Nr. 4/88

DM 14,80 · öS 124 · SFR 14,80

C16/P4-SPECIAL

C 16 - P4 SPECIAL

Kassetten auf Disketten kopieren

Hardcopy-Grundlagen

DFÜ mit dem Plus4

System-Chirurgie

Turtle-Grafik

Viele Listings

Großer Spiele-Sonder-Teil

Grüss Gott-Guten Tag

Anders als die Tageszeitung, die nur von interessanten Begebenheiten berichtet, soll die Fachzeitschrift dem Leser greifbaren Nutzen bringen. Schnell sichtbar wird dieser, wenn es sich um finanzielle Einsparungen handelt oder wenn durch geeignete Programme mehr mit dem Computer anzufangen ist. Etwas mühsamer, doch im Endeffekt ergiebiger, ist die Aneignung von Wissen und die Fähigkeit, das Wissen in die en Tat umzusetzen.
Unsere Zeitschrift für den C16, C116 und den Plus/4 nützt Ihnen in vielerlei Hinsicht. Machen Sie sich ein Bild, was Sie unter anderem in diesem Heft erwartet.
Mehr mit dem Computer anfangen zu können, ist der Wunsch vieler Benutzer. Weil schnell etwas sichtbar wird, erfreuen grafische Entwürfe sich außerordentlicher Beliebtheit. CAD- und Malprogramme unterstützen zwar kreatives Gestalten, bieten aber meist nicht die Möglichkeiten einer grafischen Programmiersprache. Drehen, Vergrößern und Verkleinern von entworfenen Figuren sind Optionen von meist sehr teuren Programmen.
Ideal sind die Möglichkeiten, die die Turtle-Grafik einer Programmiersprache wie LOGO bietet.
In Unterprogrammen lassen sich Figuren, wie zum Beispiel eine Blume, definieren. Durch Angabe von Koordinaten, Winkel und Längen, erscheinen die Figuren an beliebiger Stelle, in beliebiger Größe und beliebig gedreht auf dem Bildschirm. Programmschleifen sorgen für eine beliebige Anzahl von Aneinanderreihungen der Figuren. Nicht die Schildkröte, englisch Turtle, zeichnet im vorliegenden Programm IGELGRAFIK die Figuren. Wie der Name bereits vermuten läßt, nennt sich die Schildkröte in diesem Programm Igel. Deutsche Befehlsworte wie vorwärts und rückwärts, links und rechts steuern den Lauf dieses kybernetischen Tieres. Schön wäre es, die auf dem Bildschirm erscheinenden Grafiken auch auf jeden Drucker zu bekommen.
Obwohl für Commodore-Drucker bereits im C16-P/4-SPECIAL eine HIRES-Hardcopyroutine erschien, verstummten nicht die Stimmen unserer Leser, die gerne mehr über Hardcopies wüßten. Nicht jeder am Commodore-Rechner zu betreibende Drucker ist auch in punkto Grafik Commodore-kompatibel. Da nicht auf die Besonderheiten hunderter Drucker eingegangen werden kann, schien ein Grundlagenartikel die richtige Lösung zu sein. Sie erfahren in unserem Heft, wie der Grafikbildschirm organisiert ist und wie die Information konvertiert werden muß, um den Bildschirminhalt auch dem Drucker richtig zu übergeben.
Sie finden Listings für Commodore-, Epson-, IBM- und sonstige Drucker detailliert erläutert mit zusätzlichen Informationen, wie diese abgeändert werden können, um einen etwas anderen Ausdruck zu erreichen.
Auch für nicht grafikfähige Drucker gibt es einen Weg, mit dem Grafiken ausgedruckt werden können. Warum ein Kreis nicht immer ein Kreis wird, und wie er doch kreisrund auf dem Drucker wiedergegeben werden kann, erfahren Sie aus dem Beitrag in diesem Heft. Weiter geht es dann im nächsten Heft mit Hardcopies in verschiedenen Größen und Dichten.
Nach den Beiträgen zum Befehlssatz der CPU wurde der Wunsch laut, auch kommentierte Anwendungsbeispiele für Maschinensprache zu sehen. Mit einer ganz besonderen Technik befaßt sich der Artikel SYSTEMCHIRURGIE. Durch Verbiegen von sogenannten Sprungvektoren wird der Rechner veranlaßt, bestimmte Funktionen in etwas anderer als der gewohnten Weise auszuführen. So läßt sich der eingebaute BASIC-Editor auch als Texteditor benutzen, das Cursorblinken ausschalten und die ständige Ausgabe der Systemzeit, unabhängig von etwaigen laufenden Programmen, bewirken. Die Anwendungsmöglichkeiten für diese Technik sind zahllos. Viele Wünsche, die BASIC nicht erfüllt, lassen sich verwirklichen. Experimentieren lohnt sich.
Der Plus/4 mit seinem Userport und seiner RS-232-Schnittstelle eignet sich ideal zum Anschluß eines Modems oder eines Akustikkopplers. Der Plus/4 ist wohl einer der preiswertesten Rechner für die Datenfernübertragung. Deshalb beginnen wir jetzt auch über DFÜ zu berichten. Ein DFÜ-Lexikon vermittelt die erforderlichen Grundbegriffe.
Mit nur zwei POKEs und einem SYS lassen sich sequentielle Dateien in Programme umwandeln. Ob es sich um per DFÜ empfangene Listings oder um mit den Möglichkeiten einer Textverarbeitung überarbeitete Programme handelt, Watson und Holmes lösen in einer Geschichte den Fall auf ziemlich unkonventionelle Art.
Klar, daß wir die Reihe über Programmierung in BASIC fortsetzen.

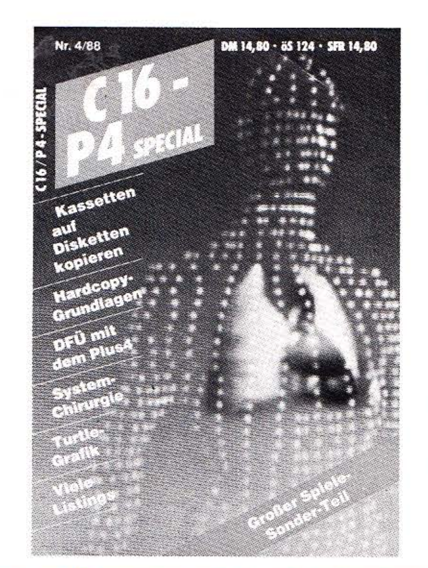
Ihre C16/P4-Crew

INHALT

GRUNDLAGEN

SERIE & SERVICE

TIPS & TRICKS



STÄNDIGE RUBRIKEN

LISTINGS

ANWENDUNGEN

SPIELE

LOAD & RUN

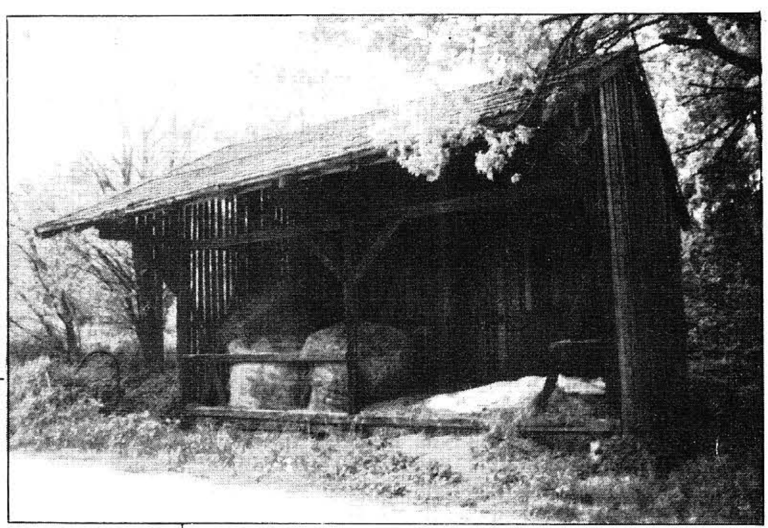
Unter Dach und Fach

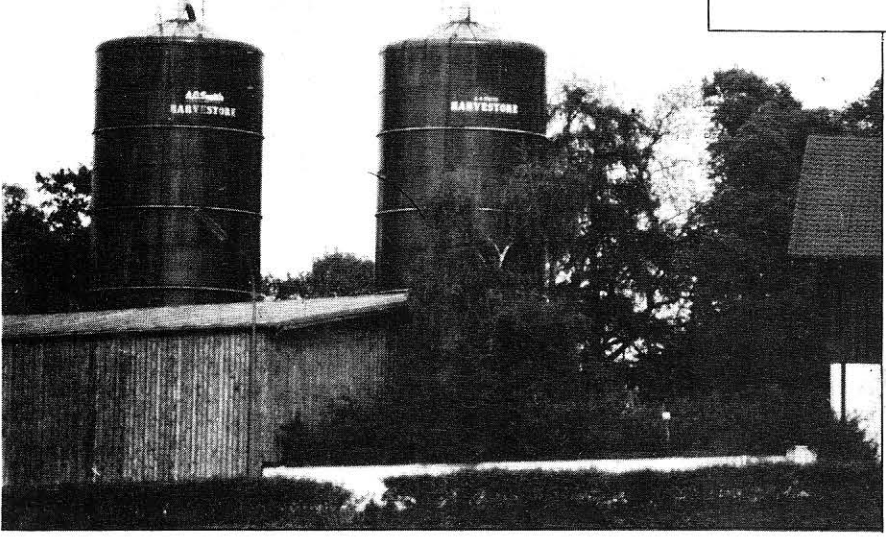
Wer mit Computern arbeitet, muß Daten und Programme speichern können. Externe Speicher kennt jeder: Diskette, Magnetband und Plattenspeicher. Aber auch der Rechner selbst braucht Speichermöglichkeiten: Arbeitsspeicher, Zwischenspeicher, Festwertspeicher und andere. Diese auf der Platine des Rechners untergebrachten Speicher sind alle in Chips untergebracht.

1860

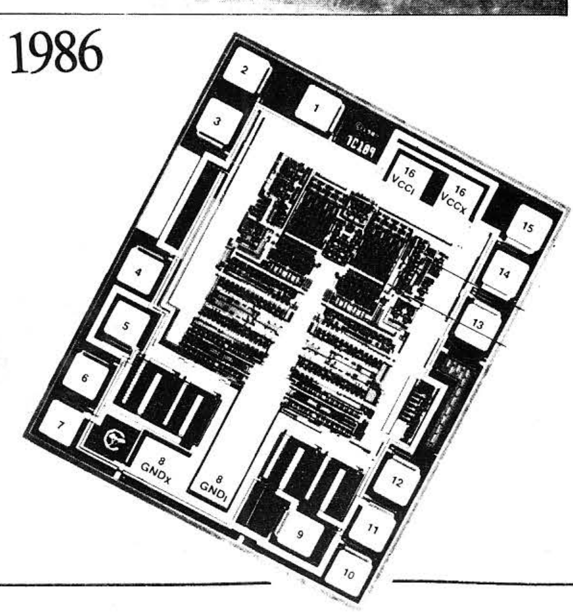
Entwicklungsgeschichte der Massenspeicher

1965





1986



Unter Dach und Fach

Programme und Daten werden in binär kodierter Form abgespeichert. Jeweils acht Bit werden zu einem Speicherplatz zusammengefaßt und erhalten eine Adresse. Eine spezielle Adressierlogik macht es möglich, jeden Speicherplatz einzeln anzusteuern. Dabei ordnet der Speicher einer vorgegebenen Adresse das jeweilige Datenwort (acht Bit) zu.
Die Anzahl der Gesamtbit eines Speichers ist immer eine Potenz von 2.

2^{10} Bit = 1024 Bit = 1 KBit
2^{11} Bit = 2048 Bit = 2 KBit
2^{15} Bit = 32768 Bit = 32 KBit
2^{16} Bit = 64536 Bit = 64 KBit

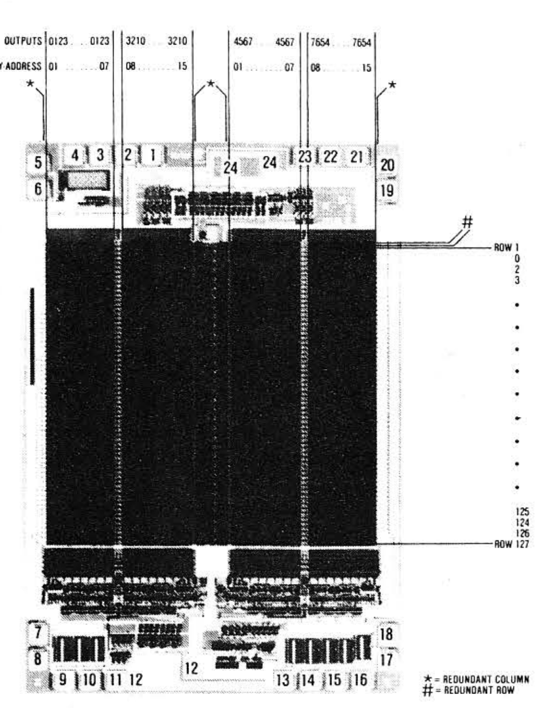
Bekanntlich ist das Bit (Binary Digit) die kleinste Darstellungseinheit für binäre Daten (0 oder 1). Um eine größere Anzahl von alphanumerischen Zeichen, etwa das Alphabet und die Ziffern von 0 bis 9, sowie Sonder- und Steuerzeichen darstellen zu können, werden jeweils acht Bit zu einer Einheit zusammengefaßt: zu einem Byte. Ein Byte ermöglicht so die Darstellung von 256 verschiedenen Zeichen. 256 ist die Anzahl der möglichen Kombinationen von 0 und 1, wenn insgesamt acht Stellen zur Verfügung stehen.
In der Praxis wird zu den acht Bit noch ein sogenanntes Prüfbit „gepackt" (sodaß es eigentlich neun Bit sind).
Die acht Datenbit können auf drei unterschiedliche Arten belegt werden:

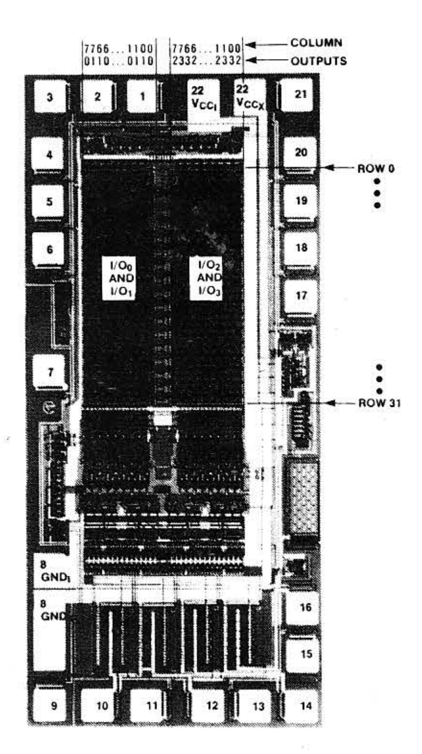
Binäre Darstellung.
Hier können Dezimalzahlen von 0 bis 255 (=256 Stück) in einem Byte gespeichert werden.
Dies ist im allgemeinen die günstigste Darstellungsart und bedingt die günstigste Speicheraufteilung.

Numerische Darstellung.
Hier wird ein Byte in zwei Tetraden unterteilt. Im sogenannten „gepackten Format" lassen sich so zwei Dezimalstellen verschlüsseln.

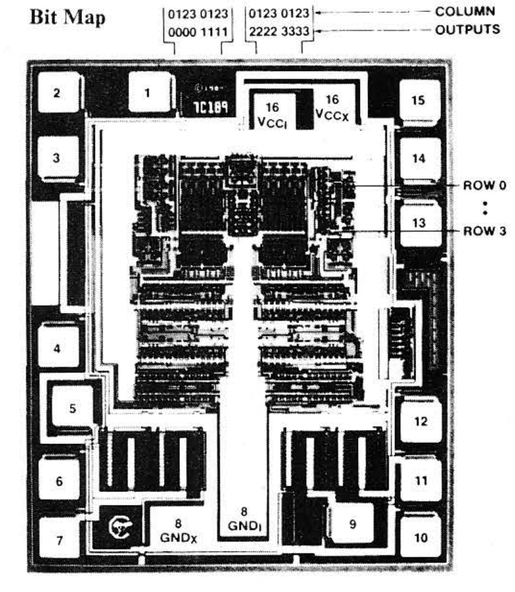
Alphanumerische Darstellung.
Wie bei der numerischen Darstellung wird auch hier das Byte in zwei Zonen von je vier Bit unterteilt: Die rechte Hälfte des Byte ist das niedrigwertige Bit und die linke Hälfte das höherwertige Bit. Dezimalziffern werden nur in den niedrigwertigen Bit verschlüsselt, während Buchstaben und Sonderzeichen durch die Kombination einer Ziffernverschlüsselung (s. o.) und einer Zonenverschlüsselung in den übrigen Bit vorgenommen wird. Speichergrößen werden daher nicht in Bit, sondern in Byte angegeben:

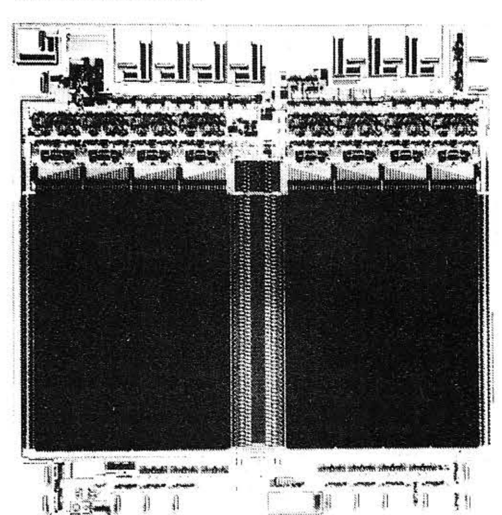
2^{19} Bit = 524.288 Bit = 512 KBit = 64 KByte
2^{20} Bit = 1.048.576 Bit = 1.024 KBit = 128 KByte
2^{21} Bit = 2.097.152 Bit = 2.048 KBit = 256 KByte
2^{22} Bit = 4.194.304 Bit = 4.096 KBit = 512 KByte





Das Layout der verschiedenen Speicherchips ist sehr ähnlich. Die verschiedenen Technologien sind nicht zu unterscheiden.





Die dunklen Flächen zeigen tausende von Speicherzellen.

RAM

Beim RAM (Random Access Memory) handelt es sich um einen Speicher mit wahlfreiem Zugriff. Er kann wahlweise ausgelesen oder mit neuen Daten beschickt werden. Aus diesem Grunde spricht man auch von einem Schreib- und Lesespeicher. Wahlfrei bedeutet aber auch, daß man sich Daten (über Adressen) auswählen kann, im Gegensatz zu einem Speicher, der nur den kompletten Inhalt freigibt. RAM-Speicher ermöglichen einen direkten Zugriff durch direkte Adressierung.
Eine weitere Eigenschaft zeichnet das

RAM aus. Nach dem Einschalten des Rechners enthält der RAM-Speicher irgendeine zufällige Information, also nicht nur Nullen. Das resultiert aus der Flüchtigkeit dieses Speichermediums: Beim Abschalten des Rechners verliert er die gespeicherten Informationen und befindet sich somit in einem undefinierten Zustand, man kann also auch nicht sagen, daß er keine Informationen enthält, eben nur ungeordnete.

Das RAM benötigt zum Erhalten der auf ihm gespeicherten Information elektrische Energie. Durch die Entwicklung extrem stromsparender RAM-Bausteine wurde es möglich, diese bei ausgeschaltetem Rechner mit Batteriestrom zu versorgen. Dies bedeutet, die Information bleibt erhalten, solange der Ladezustand der Batterie es erlaubt. Bei erneutem Einschalten des Rechners wird die Batterie nachgeladen. Man spricht hier von Batteriepufferung.

ROM

Wie der Name ROM (für Read Only Memory) schon sagt, können hier nur Daten gelesen werden. Weil der Informationsinhalt des ROM nicht verändert werden kann, spricht man auch von einem Festwertspeicher. ROM-Chips haben eine wichtige Funktion im Rechner: Sie enthalten beispielsweise Betriebssystem, Programmiersprachen oder eine Laderoutine (für das Betriebssystem auf der Diskette und anderes). Beim Einschalten des Rechners wird ein Programm von einem ROM gebootet (also automatisch geladen). Dieses Programm ermöglicht erst den Dialog mit dem Computer (über Tastatur, Bildschirm). Meist ist dieses Programm entweder ein Lader für das Betriebssystem oder das Betriebssystem selbst.

Der Inhalt des ROM ist unveränderbar und wird auch beim Ausschalten des Rechners nicht gelöscht. Zum Erhalten der gespeicherten Information ist keine Energie notwendig.

Festwertspeicher sind wegen der nicht notwendigen Adressierung sehr schnell und leicht in Systeme zu integrieren.

PROM

Das PROM (Programmable Read Only Memory) kann vom Anwender nur einmal programmiert werden. Die Informationen werden eingebrannt und können dann nicht mehr geändert werden. Dieser programmierbare Festwertspeicher verhält sich daraufhin genau wie ein ROM.

Ob ein ROM oder ein PROM in ein Gerät eingebaut wird, hängt im wesentlichen von der produzierten Stückzahl des Gerätes ab. Bei großen Serienfertigungen wird man meist ROMs verwenden, deren Entwicklung zwar sehr teuer ist, die Massenproduktion aber billig wird. PROMs müssen einzeln programmiert werden, daher eignen sie sich mehr für kleine Serien oder sogar Einzelanfertigungen.

Wie das PROM kann auch das EPROM (Erasable Programmable Read Only Memory) vom Anwender programmiert und dann wie ein ROM benutzt werden. Das EPROM ist ein löschbarer Speicher. Mit Hilfe von UV-Licht kann eine Programmierung rückgängig gemacht, das EPROM gelöscht werden. Daraufhin ist es wieder bereit, neue Informationen aufzunehmen und zu speichern. Daher eignen sich EEROMs hervorragend für Software-Entwicklungen. Meist wird es nicht fest auf Platinen verlötet, sondern nur in IC-Sockel gesteckt, um es zum Löschen und erneuten Programmieren von der Platine wieder entfernen zu können.

Der EPROM-Chip besitzt auf seiner Oberfläche ein Fenster für den Löschvorgang. Dies unterscheidet ihn schon rein äußerlich von anderen Chips.

Zum Löschen des Speicherinhaltes wird der Chip unter eine spezielle Löschlampe (UV-Licht) gebracht.

EEPROM

EEPROM (Electrical Erasable and Program Read Only Memory) ist eine Weiterentwicklung der zuvor beschriebenen EPROM-Chips. Im Gegensatz zu diesen müssen sie nicht mit UV-Licht gelöscht werden, um neu programmiert werden zu können. Einzelne Speicherzellen können bei erneuter Programmierung einfach überschrieben werden. Sie sind bedienungsfreundlicher als EPROMs und zeichnen sich zudem durch eine höhere Lebensdauer aus, durch den Wegfall der möglichen UV-Bestrahlung.

Dipl. Ing. (FH) Oliver Rosenbaum

Suchverfahren in der EDV

GESUCHT – GEFUNDEN!

Hat man es mit größeren Datenmengen oder großen Dateien zu tun und will man rationell, also schnell mit ihnen umgehen können, stößt man sehr bald auf das Problem des Suchens. Da es verschiedene Suchverfahren gibt, muß man je nach Art der zu bearbeitenden Daten zwischen ihnen wählen.



Hier sollen nur die gebräuchlichsten Verfahren aufgezeigt werden. Es handelt sich dabei um Suchverfahren, die aus dem mathematisch-technischen Bereich stammen, sich bedingt aber auch in der Textverarbeitung anwenden lassen.

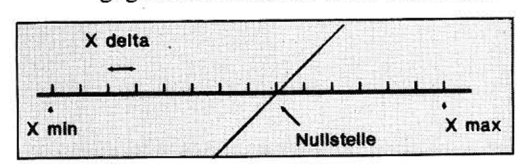
1. Sequentielles Suchen

Das einfachste „Suchen" besteht darin, jeden Wert der Datenmenge auf das Merkmal hin zu prüfen, das der gesuchte Wert aufweisen soll. Zwar führt auch diese Methode zum gewünschten Ziel, hat jedoch den Nachteil, daß sie unter Umständen sehr viel Zeit in Anspruch nehmen kann. Steht der gesuchte Wert beispielsweise an letzter Stelle in der Reihenfolge, in der geprüft wird, müssen alle Werte der Menge auf das entsprechende Merkmal hin untersucht werden.
Ein „Suchen von zwei Seiten" bringt hier auch keine Verbesserung. Man könnte etwa abwechselnd von vorne und von hinten einen Wert untersuchen.

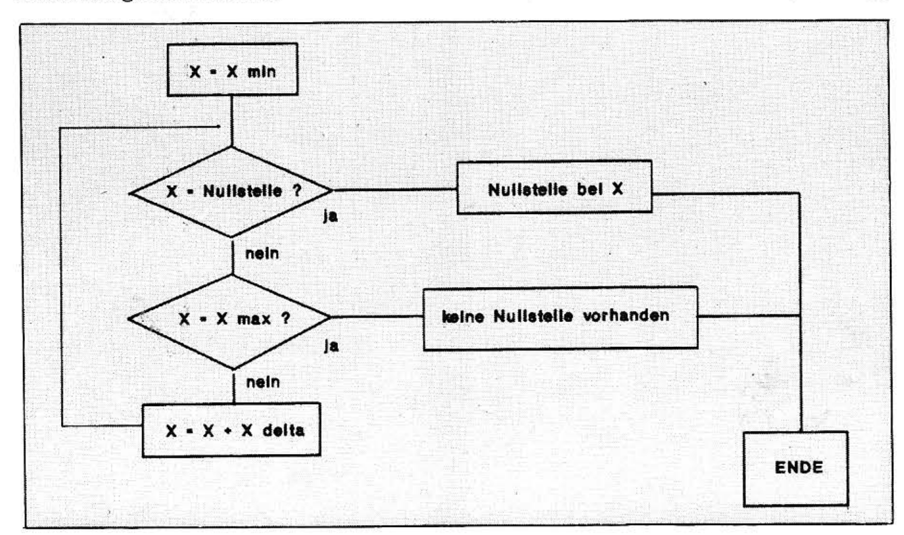
Menge	A B C D E F G H I J K
Reihenfolge der Untersuchung	1 3 5 ... ... 6 4 2

Liegt der gesuchte Wert jedoch in der Mitte, hat man den gleichen unliebsamen Nebeneffekt, alle Werte untersucht zu haben.

Beispiel:
Suchen einer Nullstelle in einem vorgegebenen Bereich einer Funktion.

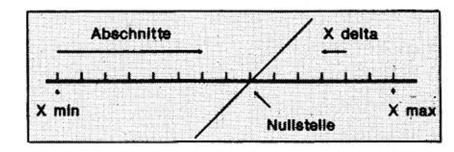


Ablaufdiagramm hierzu:

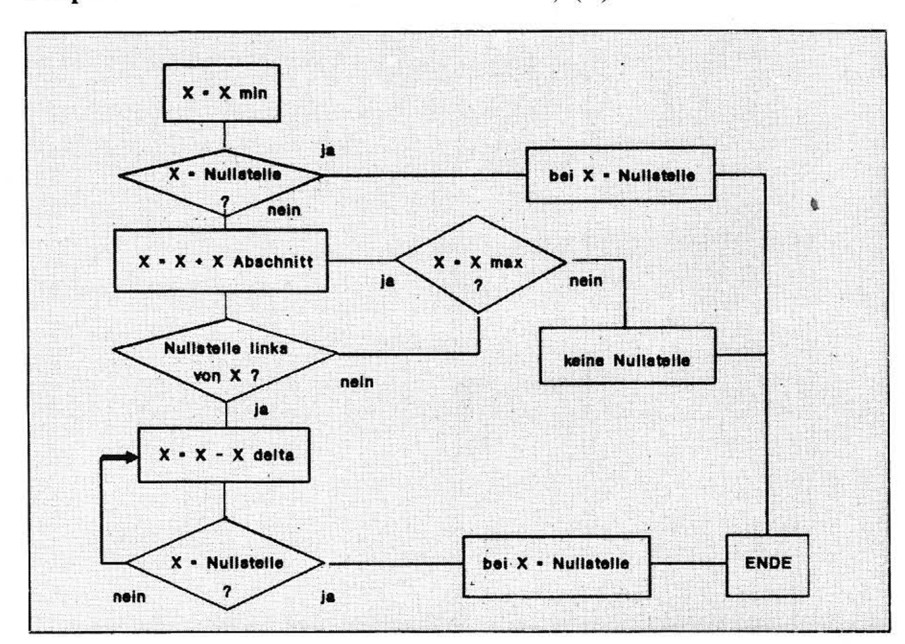


2. Sprungverfahren

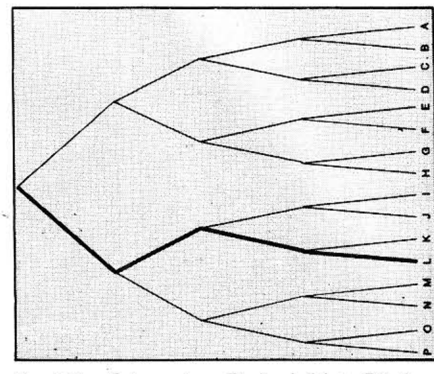
Beim Sprungverfahren wird ähnlich wie beim sequentiellen Suchen das zu untersuchende Intervall in größere Abschnitte unterteilt, die daraufhin geprüft werden, ob ein Wert mit dem gesuchten Merkmal (etwa der Nullstelle) hier enthalten ist.
Ist ein solcher Abschnitt gefunden, wird in diesem sequentiell gesucht (s. o.).



Beispiel:



Intervall, welches den gesuchten Wert enthält, wird wiederum halbiert...



3. a) Im folgenden Beispiel ist „L" der gesuchte Wert. (-) bedeutet, daß in diesem Bereich der Wert nicht vorhanden ist, (+) kennzeichnet hier das

3. Intervallsuchverfahren

Die Menge, in welcher sich der gesuchte Wert befindet, wird zunächst halbiert und es wird untersucht, ob sich der gesuchte Wert im ersten oder im zweiten Intervall befindet. Das Intervall, das ein Element mit dem Merkmal enthält, nach dem gesucht wird.
Zur Vereinfachung ist hier eine gerade Anzahl von Elementen gegeben, somit wird im letzten Intervall nur noch ein Wert übrigbleiben. (Andernfalls müßte man zusätzliche Schritte zum Abfragen der Grenzen oder der zu untersuchenden Wertemenge einfügen.

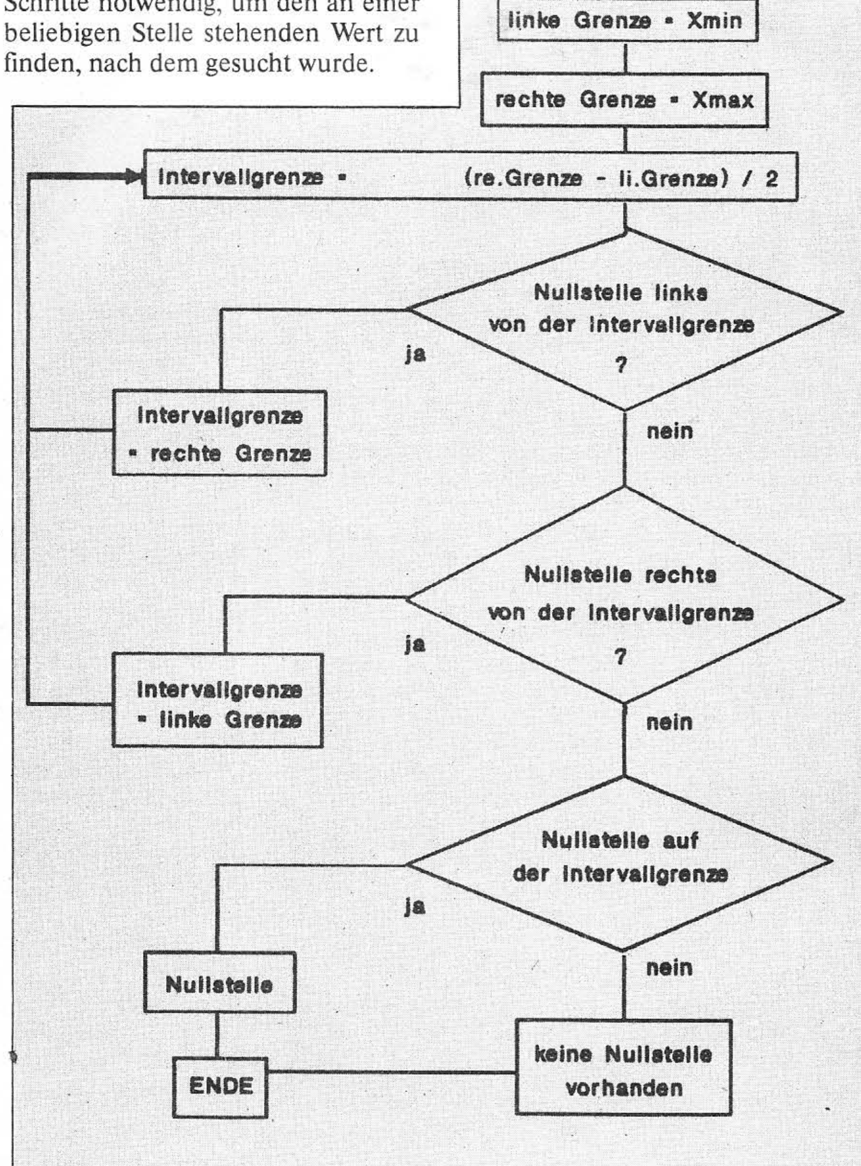
1. Suchvorgang
- - - - - - - - - - -/+ + + + + + + + +
A B C D E F G H I J K L M N O P

2. Suchvorgang
+ + + +/- - - -
A B C D E F G H I J K L M N O P

3. Suchvorgang
- -/+ +
A B C D E F G H I J K L M N O P

4. Suchvorgang
-/+
A B C D E F G H I J K L M N O P

„L“ ist hier als einzigster Wert im möglichen Wertebereich übrig geblieben und damit gefunden.
In diesem Beispiel sind also nur vier Schritte notwendig, um den an einer beliebigen Stelle stehenden Wert zu finden, nach dem gesucht wurde.



3. b) Ein weiteres Beispiel für das Intervallsuchverfahren ist die, aus der Mathematik bekannte, Intervallschachtelung:
Gesucht ist beispielsweise die Konstante Pi, auf zwei Stellen hinter dem Komma genau. Hier muß so lange das Intervall halbiert werden, bis die geforderte Genauigkeit (Stellen hinter dem Koma) erreicht worden sind.

1. Intervall	3,00	bis	4,00
2. Intervall	3,00	bis	3,50
3. Intervall	3,00	bis	3,25
4. Intervall	3,125	bis	3,25
5. Intervall	3,125	bis	3,1875
6. Intervall	3,125	bis	3,1563
7. Intervall	3,1407	bis	3,1563
8. Intervall	3,1407	bis	3,1485

Nun kann abgebrochen werden, da die zwei ersten Stellen hinter dem Komma konstant bleiben.
Der Wert für Pi kann hier mit 3,14 im Rahmen der gewählten Genauigkeit angenommen werden.

3. c) Anhand eines Fließschemas soll das Suchen der Nullstelle einer Funktion mit Hilfe des Intervallverfahrens gezeigt werden.

Zusammenfassend ist zu sagen, daß ein qualitativer Vergleich der unterschiedlichen Suchverfahren schlecht möglich ist, da die Auswahl des Suchverfahrens in sehr starkem Maße von dem zu programmierenden Problem und nicht zuletzt von dem verwendeten - meist vorgegebenen - Speichermedium abhängig ist.
Bandkassetten etwa lassen im allgemeinen nur ein sequentielles Suchen zu, da sie meist in nur einer Richtung hintereinander Daten speichern können.
Auch die Menge der zu bearbeitenden (hier zu durchsuchenden) Daten kann für die Wahl eines bestimmten Suchverfahrens ausschlaggebend sein. Bei kleineren Datenmengen ist ein gutes, also sehr schnelles Suchverfahren oft zu aufwendig und damit unwirtschaftlich.
Bei sehr großen Datenmengen ist abzuwägen, ob nicht zunächst die Datenbank nach bestimmten Gesichtspunkten sortiert werden kann, sodaß bei einem Suchdurchgang nicht die gesamte Datenmenge zur Verfügung steht, sondern nur der in Frage kommende Teil. Dies ist besonders dann von Vorteil, wenn sehr oft „gesucht“ wird.

Dipl. Ing. (FH) Oliver Rosenbaum

SYSTEM-CHIRURGIE

Eingriffe in Rechner-Routinen

Mit derselben Technik, durch die die gefürchteten Computerviren erzeugt werden, lassen sich viele nützliche Utilities programmieren. Wir zeigen Ihnen, wie dies funktioniert; sowohl im Prinzip als auch an ausgewählten Beispielen. Wollen Sie den BASIC-Editor als Texteditor benutzen, das Blinken des Cursors ausschalten oder hätten Sie gar gerne eine ständig präsente Systemuhr? Hier finden Sie alles, was Sie dazu brauchen.

Ein kleiner Systemeingriff kann bereits etwas bewirken, was sonst mit seitenlangen Programmen bewerkstelligt werden muß. So sind nur wenige Programmzeilen notwendig, um den eingebauten BASIC-Editor in einen leistungsfähigen Texteditor umzufunktionieren. Oft wird gewünscht, mit dem Rechner wie gewohnt zu arbeiten. Nur eine Kleinigkeit sollte anders sein.

Eine Uhr zu programmieren, die ständig die richtige Zeit anzeigt, ist weiter nicht schwer. Jedoch wenn Disketten formatiert, BASIC-Programme abgetippt und aufgerufen werden und vielleicht sogar noch in die Textverarbeitung SCRIPT/PLUS gegangen wird und die Uhr in der linken oberen Ecke des Bildschirms springt immer noch jede Sekunde weiter, dann mag dies manchem als Hexerei erscheinen. Dennoch ist es möglich. Das Zauberwort heißt Systemeinbindung.

Was ist nun Systemeinbindung? Wir verdeutlichen uns das am besten an einem Beispiel. Was passiert, wenn wir PRINT"U" eingeben, dürfte bekannt sein. Es erscheint der Buchstabe U auf dem Bildschirm. Jedoch muß dies nicht unbedingt sein. Genausogut hätte der Computer auch den Bildschirm löschen oder „Happy Birthday" spielen, womöglich uns gar ein X statt eines U vormachen können

Der BASIC-Interpreter und das Betriebssystem unseres Rechners bestimmen, was mit PRINT"U" geschehen soll. Wenn wir Routinen des Betriebssystems oder des BASIC-Interpreters abändern, können wir den Rechner veranlassen, ein X statt eines U auszugeben.

BETRIEBSSYSTEM IM ROM

Da das Betriebssystem im ROM liegt, kann es leider nicht geändert werden. Wenn unser Rechner 64 KByte Hauptspeicher besitzt, könnten wir den ROM-Inhalt ins RAM kopieren, wo sich nach Belieben Änderungen vornehmen lassen. Es müssen nur noch alle Umschaltungen ins ROM aus dem Betriebssystem entfernt und die RAM-Bank eingeschaltet werden. Dieser Weg ist nicht der einzig gangbare. Es gibt noch einen einfacheren, der auch mit Rechnern geringerer Speicherkapazität funktioniert. Dafür haben bereits die Systemprogrammierer gesorgt. Der Schlüssel dazu ist der indirekte Sprung. In BASIC kennen wir nur den direkten, wobei auf GOTO eine Zeilennummer folgt. Wäre statt dessen auch eine numerische Variable zulässig, könnten Sprungziele durch neue Wertzuweisungen ganz schön manipuliert werden.

Die abgebildete Sprungtabelle zeigt eine Auflistung aller indirekten Sprünge. Die linke Spalte zeigt die Adressen, an denen der indirekte Sprung steht. Darauf folgt in Assembler-Mnemonics der Sprungbefehl. Im RAM-Bereich von $0300 bis $0331 stehen die Adressen, auf die der indirekte Sprung letztendlich verzweigt. Nach dem Doppelpunkt finden wir die dort gespeicherte Sprungadresse in der Reihe Low-Byte und High-Byte. Nach dem Strichpunkt kommt die Benennung dieser Adresse mit einer kurzen, zwischen Klammern gesetzten Erläuterung.

Betrachten wir uns die Routine BSOUT, die für die Zeichenausgabe verantwortlich ist. Aufgerufen sollte sie in Maschinenprogrammen mit JSR $FFD2 werden. Mit JSR $EC4B könnten wir zwar denselben Effekt erreichen. Allerdings müßten hierbei zwei Nachteile in Kauf genommen werden: die Kompatibilität zu den anderen Commodore-Maschinen und die Möglichkeit der Routinenänderung gingen verloren.

ROUTINEN-ÄNDERUNG

Soll eine Routine durch eine neue ersetzt werden, so kann diese irgendwo an eine nicht genutzte Stelle des ungebankten RAM-Bereiches von $0000 bis $7FFF geschrieben werden. Bei 16-K-Byte-Rechnern geht es nur bis zur Speicherstelle $3FFF. Es bietet sich zu diesen Zwecken der ungenutzte Bereich von

TABELLE INDIREKTER SPRÜNGE

.8683	jmp ($0300)	:	86 86	;	IERROR	(Fehlermeldung
.870f	jmp ($0302)	:	12 87	;	IMAIN	(Editor-Schleife)
.8953	jmp ($0304)	:	56 89	;	ICRNCH	(Tokenisierung)
.8b6b	jmp ($0306)	:	6e 8b	;	IQPLOP	(Keywort)
.8bd3	jmp ($0308)	:	d6 8b	;	IGONE	(Hauptschleife)
.9414	jmp ($030a)	:	17 94	;	IEVAL	(Eval)
.8967	jmp ($030c)	:	6a 89	;	IESCLK	(User-Token)
.8b85	jmp ($030e)	:	88 8b	;	IESCPR	(User-Keywort)
.8c88	jmp ($0310)	:	8b 8c	;	IESCEX	(User-Execute)
.ce3f	jmp ($0312)	:	42 ce	;	ITIME	(Interrupt-Uhr)
.ce08	jmp ($0314)	:	0e ce	;	CINV	(Interrupt)
.ce0b	jmp ($0316)	:	4c f4	;	CBINV	(Break-Interrupt)
.ffc0	jmp ($0318)	:	53 ef	;	IOPEN	(Open)
.ffc3	jmp ($031a)	:	5d ee	;	ICLOSE	(Close)
.ffc6	jmp ($031c)	:	18 ed	;	ICHKIN	(Chkin)
.ffc9	jmp ($031e)	:	60 ed	;	ICKOUT	(Chkout)
.ffcc	jmp ($0320)	:	0c ef	;	ICLRCH	(Clrch)
.ffcf	jmp ($0322)	:	e8 eb	;	IBASIN	(Basin)
.ffd2	jmp ($0324)	:	4b ec	;	IBSOUT	(Bsout)
.ffe1	jmp ($0326)	:	65 f2	;	ISTOP	(Stop)
.ffe4	jmp ($0328)	:	d9 eb	;	IGETIN	(Getin)
.ffe7	jmp ($032a)	:	08 ef	;	ICLALL	(Clall)
.----	jmp ($032c)	:	4c f4	;	USRCMD	(Monitor Break)
.f047	jmp ($032e)	:	4a f0	;	ILOAD	(Load)
.f1a1	jmp ($0330)	:	a4 f1	;	ISAVE	(Save)

$065E bis $06EB an, der von Commodore ursprünglich für einen Sprachsynthesizer vorgesehen war, nun aber brachliegt.
Weitere Möglichkeiten sind ein Teilbereich des Prozessorstacks, des BASIC-Pseudostacks, der Kassettenpuffer bei Diskettenbetrieb, der RS-232-Input-Puffer und RAM-Bereiche von $1000 bis $7FFF, die durch Verlegen von BASIC-Anfang oder BASIC-Ende vor dem Überschreiben durch BASIC-Programme zu schützen sind.
Eine solche selbstgeschriebene Routine bewirkt nichts, wenn sie nicht eingebunden wird. Hierzu braucht nur die im Sprungvektor eingetragene Adresse durch die Anfangsadresse unserer neuen Routine ersetzt zu werden.
Um dieses gleich einmal in der Praxis auszuprobieren, lassen wir bei der Ausgabe alle Buchstaben U durch ein X ersetzen.

X STATT U

Zuerst sind die hexadezimalen ASCII-Werte für U und X zu beschaffen. Wir bekommen sie mit:

?HEX$(ASC("U"))
?HEX$(ASC("X"))

Nun gehen wir mit MONITOR in den Maschinenmonitor. Dort geben wir folgende Befehle ein:

```
A 065E CMP #$55
A 0660 BNE $0664
A 0662 LDA #$58
A 0664 JMP $ec4b
```

Der Compare-Befehl vergleicht, ob ein U vorliegt. Falls nein, geht der Sprung zur Ausgabe. Andernfalls wird $58, der ASCII-Wert für X, in den Akku geladen und dadurch der Buchstabe X ausgegeben. Zur Einbindung der neuen Routine rufen wir auf:

M324

In der ersten Zeile steht:

>0324 4B EC . . . . . .

Anstelle der Adresse $EC4B tragen wir die Anfangsadresse unserer Routine $065E ein. Hierzu gehen wir mit dem Cursor in die betreffende Zeile und ändern sie ab in:

>0324 5E 06 . . . . . .

Nach dem Druck der Return-Taste ist das Werk vollbracht. Zum Begutachten wird der Monitor mit X und Return verlassen. Ein Test gibt Aufschluß über die gelungene Manipulation.

PRINT"TUV"

Der Computer gibt auf den Bildschirm die Zeichen TXV aus.

SELEKTION NACH GERÄTENUMMER

Mit der gezeigten Methode lassen sich auch sinnvollere Anwendungen programmieren, wie zum Beispiel eine Druckeranpassung oder die Umwandlung eines im Standard-ASCII-Format auf Datenträger vorliegenden Textfiles in das CBM-ASCII-Format. Hierzu sind folgende Adressen von Nutzen:

```
>0098 ; DFLTN
        (Eingabegerät)
>0099 ; DFLTO
        (Ausgabegerät)
```

Durch Abfrage des Ausgabegerätes kann festgestellt werden, ob das Zeichen an den Drucker gehen soll. Wir tauschen als Beispiel U gegen X nur bei der Druckerausgabe. Zuerst ist der noch verbogene Ausgabezeiger auf den ursprünglichen Wert zu setzen. Dies erreichen wir durch einen Reset oder, in unserem Falle noch besser, durch Drücken des Resetknopfes, während die Run/Stop-Taste gedrückt ist. In diesem Falle sind wir bereits im Monitor und können eingeben:

```
A 065E PHA
A 065F LDA$99
A 0661 CMP #$04
A 0663 BNE $066D
A 0665 PLA
A 0666 CMP #$55
A 0668 BNE $066E
A 066A LDA #$58
A 066C PHA
A 066D PLA
A 066E JMP $EC4B
```

Das auszugebende Zeichen wird durch PHA auf den Stack gesichert, damit es bei der Abfrage des Ausgabegerätes nicht verlorengeht. Ist nicht Gerät Nummer vier, also der Drucker, das Ausgabegerät, erfolgt der Sprung nach $066D, wo das Zeichen wieder vom Stapel geholt und anschließend ausgegeben wird. Andernfalls holen wir uns das Zeichen mit PLA vom Stapel und überprüfen, ob es sich um ein U handelt. Wenn ja, wird ein X auf den Stapel geschoben.
Bei den abgeänderten Routinen darf nicht die Adresse, von der aus der Sprung kommt, verwendet werden. Bei JMP $FFD2 am Ende dieses Listings würde sonst diese Routine in einer Endlosschleife ständig durchlaufen werden. Ergebnis wäre, daß sich nichts mehr rührt.

BASIC-EDITOR ERFASST TEXTE

Nachdem die neue Routine fertiggestellt ist, binden wir die Anfangsadresse genau wie vorhin wieder in das System ein. In BASIC bringt PRINT"TUV" ganz normal die Buchstaben TUV auf den Bildschirm. Anders verhält sich der Drucker:

OPEN,4:PRINT#4,
"TUV":CLOSE4

Gedruckt werden die Buchstaben TXV.
Im normalen Zustand erweist sich der BASIC-Editor als ungeeignet für Texte. Mit CBM- und Shift-Taste schalten wir in den Kleinschreibmodus um und geben ein:

1 aAbBcC
list

Großbuchstaben mag der BASIC-Editor anscheinend nicht. Ist er nicht willig, so versuchen wir es mit Gewalt:

1 remBCD
list

Er hat die Großbuchstaben angenommen. Beim Auflisten werden sie als Token angesehen und enttokenisiert, was für uns nicht annehmbar ist. Doch kommen die Großbuchstaben zweimal in der Zeichentabelle vor. Im Rechnerhandbuch steht, daß die Zeichen der Codes 192 bis 223 identisch seien mit denen von 96 bis 127. Betrachten wir die eingegebene Zeile mit dem Monitor.

monitor
m1005

Wir sehen:

>1005 8f c2 c3 c4 . . . . . .

Wir ersetzen die Hexwerte $C0 bis $DF durch $60 bis $7F. Das REM-Token $8F könnten wir zusätzlich noch in den Buchstaben A umwandeln. Wir ändern die ersten vier Werte ab Adresse $1005.

>1005 61 62 63 64

Nach der Rückkehr in den BASIC-Editor zeigt der Rechner beim Listen:

1 ABCD

Großbuchstaben können also erfaßt werden, wenn wir deren ASCII-Wert wandeln. Durch Manipulation der Eingaberoutine ist dies nicht schwer zu erreichen.

Nachdem wir dafür gesorgt haben, daß verboge-

ne Sprungvektoren wieder richtiggestellt sind, geben wir im Monitor ein kurzes Listing ein.

```
a 065e jsr   $ebe8
a 0661 bcs   $066e
a 0663 cmp   #$e0
a 0665 bcs   $066d
a 0667 cmp   #$c0
a 0669 bcc   $066e
a 066b sbc   #$60
a 066d clc
a 066e rts
```

Durch den Aufruf JSR $EBE8 lesen wir das Zeichen vom Eingabegerät. Wenn ein I/O-Error vorliegt, ist anschließend das Carry-Flag gesetzt. In diesem Fall lassen wir alles, wie es ist, und sorgen für den Rücksprung. Sonst wird verglichen, ob das empfangene Zeichen kleiner als $E0 und größer oder gleich $C0 ist. Gegebenenfalls ist nur noch der Wert $60, dezimal 96, zu subtrahieren.
Die Anfangsadresse dieser Routine ist nun, da es sich um die BASIN-Routine handelt, in die Speicherstellen $0322 und $0323 einzutragen. Es zeigt sich, daß der BASIC-Editor jetzt Großbuchstaben annimmt. Der große Vorteil dieses Verfahrens ist, daß mit dem modifizierten Editor weiterhin auch BASIC-Programme abgetippt werden können.
Nur ist in Kauf zu nehmen, daß BASIC-Befehle nicht mehr abgekürzt werden dürfen. Die neuen Großbuchstaben werden nicht mehr als Markierung für die Abkürzung erkannt. Ärgernis bereitet das Fragezeichen. Es wird weiterhin als PRINT wiedergegeben.
Ändern läßt sich dies durch die im C16/P4-SPECIAL Nr. 1 veröffentlichte Zeichensatz-Anpassung. Durch Umdefinieren des Codes für das Fragezeichen und Erstellen der Zeichenmatrix für diesen neuen Code ist die Sache erledigt. Soll auch eine Druckeranpassung erfolgen, so ist die Anfangsadresse des Maschinensprachteiles um 17 zu erhöhen, wenn Sie es nicht vorziehen, die soeben erstellte BASIN-Modifizierung zu verlegen.

INTERRUPTGESTEUERTE UHR

```
10 rem systemuhr===============c16 <ha>
20 rem (p) commodore welt team     <ho>
30 rem ==========================  <ng>
40 rem (c) by alfons mittelmeyer   <cg>
50 rem                             <pd>
60 rem                             <ah>
70 rem basic v3.5                  <nl>
80 rem c16/116/plus4               <ki>
90 rem ==========================  <jg>
100 rem anpassungsteil             <pc>
110 rem --------------             <li>
120 z=0 :rem zeile                 <jp>
130 s=0 :rem spalte                <pj>
140 nl=48+128:rem '0'              <eg>
150 tr=58+128:rem ':'              <ed>
160 rem ========================== <fh>
170 rem maschinenroutine           <kc>
180 rem ----------------           <bd>
190 fori=1630to1735                <nd>
200 reada:pokei,a:next             <oh>
210 data 198,215,208,041,169,050   <bn>
220 data 133,215,162,002,032,142   <bk>
230 data 006,202,016,250,165,208   <kh>
240 data 201,036,208,003,232,134   <lg>
250 data 208,162,002,160,007,032   <cc>
260 data 163,006,202,016,250,165   <cj>
270 data 212,160,002,145,213,160   <pc>
280 data 005,145,213,076,066,206   <di>
290 data 181,208,248,024,105,001   <fp>
300 data 216,149,208,201,096,240   <pd>
310 data 003,162,000,096,169,000   <bp>
320 data 149,208,096,181,208,041   <mg>
330 data 015,005,211,145,213,181   <po>
340 data 208,074,074,074,074,041   <ai>
350 data 015,005,211,136,145,213   <do>
360 data 136,136,096,120,169,094   <nf>
370 data 141,018,003,169,006,141   <gb>
380 data 019,003,088,096           <mm>
390 rem ========================== <ej>
400 rem initialisierung            <di>
410 rem ---------------            <lm>
420 poke215,50:poke211,nl          <oj>
430 poke212,tr:ad=3072+z*40+s      <ko>
440 h=int(ad/256):l=ad-256*h       <jf>
450 poke213,l:poke214,h            <fk>
460 scnclr:print"zeiteingabe"      <pf>
470 print"-----------":print       <ce>
480 input"hh/mm/ss";u$             <bh>
490 fori=0to2                      <gh>
500 h=val(mid$(u$,3*i+1,1))        <gf>
510 l=val(mid$(u$,3*i+2,1))        <ia>
520 poke208+i,16*h+l:next:print    <ei>
530 print"start bei tastendruck!"  <ln>
540 getkeya$:sys1723:scnclr        <oi>
550 rem ========================== <pp>
560 rem  p r o g r a m m e n d e   <hp>
570 rem ========================== <mo>
```

EINIGE PROBLEMFÄLLE

Der BASIC-Interpreter hat so seine Tücken. Was er nicht mag, sind:

- Leerzeilen;
- führende Leerzeichen;
- Zahlen am Zeilenanfang.

Jedoch kann auch hier für Abhilfe gesorgt werden. Bei der Eingabe eines geshifteten Leerzeichens werden weitere normale Leerzeichen oder Zahlen angenommen. Eine Leerzeile kann durch die Eingabe eines Leerzeichens realisiert werden. Da BASIN auch Leerzeichen am Zeilenende verwirft, ist ein Trick anzuwenden: Nach dem geshifteten Leerzeichen geben wir ein reverses ein. Beim Auflisten ist tatsächlich eine Leerzeile sichtbar.

TEXTEDITOR MIT DRUCKERAUSGABE

Beim Ausgeben von Text auf den Drucker sollen keine Zeilennummern erscheinen. Durch Modifikation der BSOUT-Routine können wir dafür sorgen. Wenn wir uns nicht daran gewöhnen wollen, daß wir für das Handling unseres Rechners keine Abkürzungen mehr verwenden können, bietet sich eine andere Lösungsmöglichkeit an: die Unterbindung der Tokenisierung bei nicht eingegebener Zeilennummer. Wenn die BASIN-Routine jedes Leerzeichen durch ein geshiftetes Leerzeichen ersetzt, so fällt in den Problemfällen, führende Leerzeichen und Zahl am Zeilenanfang, das zusätzliche Eintippen dieses Zeichens weg. Wir stellen jetzt eine neue Lösung mit mehr Know-how vor.
Dieses Assemblerlisting

brauchen Sie nicht abzutippen, da wir es für Sie noch als BASIC-Listing bereitgestellt haben.

MINI-EDITOR
Assembler-Listing

```
a 065e lda  #$00
a 0660 sta  $08
a 0662 jsr  $ebe8
a 0665 bcs  $066e
a 0667 cmp  #$20
a 0669 bne  $066d
a 066b lda  #$a0
a 066d clc
a 066e rts
a 066f ldy  $08
a 0671 beq  $0686
a 0673 jsr  $0479
a 0676 cmp  #$a0
a 0678 bne  $067d
a 067a jsr  $0473
a 067d jsr  $04a5
a 0680 iny
a 0681 cmp  #$00
a 0683 bne  $067d
a 0685 rts
a 0686 jmp  $8956
a 0689 pha
a 068a lda  $99
a 068c cmp  #$03
a 068e beq  $06ab
a 0690 lda  $06b4
a 0693 cmp  #$0d
a 0695 beq  $069c
a 0697 pla
a 0698 pha
a 0699 jsr  $ec4b
a 069c pla
a 069d cmp  #$0d
a 069f beq  $06a5
a 06a1 cmp  #$20
a 06a3 bne  $06a8
a 06a5 sta  $06b4
a 06a8 clc
a 06a9 bcc  $06af
a 06ab pla
a 06ac jsr  $ec4b
a 06af lda  #$fc
a 06b3 rts
```

EINGABEROUTINE

Die BASIN-Routine von $065E bis $066E tauscht das Leerzeichen S20 gegen das geshiftete Leerzeichen $A0. Außerdem wird in die Speicherstelle $08 eine Null geschrieben. Die Speicherstelle $08 enthält nämlich bei der Tokenisierung die Ziffernanzahl der eingegebenen Zeilennummern. Wurde keine eingegeben, so kann sie auch einen von Null verschiedenen Wert enthalten. Mit unserer Voreinstellung aber kann die Speicherstelle $08 als Flag für die eingegebene Zeilennummer dienen.
Von $066F bis $0685 geht die neue Tokenisierungsroutine CRNCH. Bei der Direkteingabe von Befehlen erfolgt eine normale Tokenisierung. Sonst wird diese unterbunden. Mit JSR $0479, CHRGOT, wird das erste Zeichen nach der Zeilennummer abgefragt.

TOKENISIERUNG

CHRGOT liest das Zeichen, auf welches der Zeiger in den Adressen $3B und $3C weist.
Da dabei die RAM-Bank eingeschaltet wird, kann der Zugriff auch auf den gebankten RAM-Bereich erfolgen. Liegt ein geshiftetes Leerzeichen vor, so beseitigen wir dieses, indem wir den Zeiger mit CHRGET, JSR $0473, weiterstellen. CHRGET macht dasselbe wie CHRGOT, nur daß zuvor zum Lesen des nächsten Zeichens der Zeiger erhöht wird.
Normale Leerzeichen werden bei diesen Lesevorgängen übersprungen. Der Interpreter bekommt dadurch bei der Abarbeitung Leerzeichen zwischen den BASIC-Befehlen gar nicht erst zu Gesicht und kann somit schneller arbeiten. Die Tokenisierungsroutine unterschlägt geshiftete Leerzeichen. Normale Leerzeichen jedoch bleiben erhalten, weil CRNCH sie gar nicht erst zu sehen bekommt und hinauswerfen kann.
Ziffern, die durch normale Leerzeichen getrennt sind, werden wegen CHRGET als zu einer einzigen Zahl gehörig angesehen. Wichtig zu wissen ist, daß die beiden Routinen das Y-Register auf Null setzen. Die CRNCH übergibt nach der Tokenisierung dem aufrufenden Programm die Zeilenlänge im Y-Register. Wir müssen daher ebenfalls die Länge der Zeile vom augenblicklichen Zeigerstand bis zu dem mit einer Null markierten Zeilenende bestimmen.
Zum Zeichenlesen dient die Routine JSR $04A5, die dasselbe macht wie LDA ($3B),Y. Der Unterschied ist, daß der Zugriff immer auf das RAM erfolgt. Nach Erreichen des Zeilenendes können wir die Routine beenden. Das aufrufende Programm bekommt die Information, was aus dem BASIC-Eingabepuffer in das Listing zu übernehmen ist.

AUSGABEROUTINE

Eine Modifizierung von BSOUT ist nicht nur wegen unserer Druckerausgabe erforderlich. Beim Listen von Zeilen mit Großbuchstaben würden diese als BASIC-Schlüsselwörter erscheinen. Dies läßt sich jedoch sehr einfach unterbinden. In die Speicherstelle $0F ist nur eine Zahl von $80 bis $FF zu schreiben. Die Listroutine behandelt dann alles so, als würde es als Text zwischen Anführungszeichen stehen, und unterläßt die Wandlung in Schlüsselwörter.
Unsere Ausgaberoutine von $0689 bis $06b3 nimmt wegen einer Sonderbehandlung für die Ausgabe auf externe Geräte relativ viel Raum in Anspruch. Damit das Ausgabegerät abgefragt werden kann, ist zuerst mit PHA das auszugebende Zeichen zu retten.
Die Bildschirmausgabe geht normal vonstatten. Bei anderen Geräten wollen wir die Zeilennummer unterdrücken. Als Flag dient die Speicherstelle $06B4, die sich unmittelbar hinter unserem Programm befindet. Zu Beginn steht darin der Wert $0D, zum Zeichen dafür, daß nichts ausgegeben werden soll. Sonst würde die Ausgabe mit JSR $EC4B erfolgen.
Was auch immer geschieht, im Anschluß wird jedenfalls abgefragt, ob es sich um ein Return mit dem Code $0D oder um das der Zeilennummer folgende Leerzeichen $20 handelt, das beim Auflisten automatisch erzeugt wird. Trifft eines von beiden zu, so wandert entweder $0D oder $20 in unsere Flag-Speicherstelle. Die Zeichen nach Return werden somit unterschlagen, die Zeichen nach Space ausgegeben.
Das BASIC-Listing MINI-EDITOR erzeugt den beschriebenen Maschinencode und stellt die drei Zeiger ICRNCH, IBASIN und IBSOUT um. Am besten schalten Sie mit AUTO10 gleich die automatische Generierung der Zeilennummern ein. Sie brauchen somit auf Zahlen am Zeilenanfang nicht mehr zu achten. Wo vorher ein geshiftetes Leerzeichen nötig war, genügt jetzt ein normales, da ja sowieso eine Umwandlung stattfindet.

BESONDERHEITEN VON MINI-EDITOR

Das Einfügen von Leerzeilen vollzieht sich auch jetzt mit dem reversen Leerzeichen. Sie können DSAVE, DLOAD und sonstige Befehle wieder abkürzen. Erfassen Sie doch einmal ein paar Zeilen und geben dann ein:

open4,4,7:cmd4:list:print#4:close4

Mit Diskette können Sie probieren:

open8,8,8,"text,s,w":cmd8:list:print#8:close8

Der Drucker druckt die Zeilennummern nicht mit. Wenn Sie mit SCRIPT/PLUS den erzeugten sequentiellen File „text" einladen, so sehen Sie, daß auch dort das näm-

liche der Fall ist. Eine kleine Macke kommt zutage, wenn auf den Großbuchstaben U unmittelbar eine Zahl folgt, die nicht durch ein Leerzeichen abgetrennt ist.
U ist das Token für das Schlüsselwort ELSE. Beim Renumbern wird die nachfolgende Zeile als Zeilennummer angesehen und ersetzt. Da den anderen Schlösselwörtern, denen ebenfalls eine Zahl folgt, keine sichtbaren Zeichen entsprechen, tritt der genannte Effekt nicht auf.

NICHTBLINKENDER CURSOR

Es existiert beim C16/116/P4 kein Poke, um das Cursorblinken abzuschalten. An die Betriebssystem-Routine, die den Cursor ein- und ausschaltet, ist auch nicht heranzukommen. Dennoch gibt es einen Weg: Es ist der Interruptvektor ITIME. Alle fünfzigstel Sekunden wird dieser Interrupt aufgerufen. Wenn wir in ihn eine Routine einbinden, können wir den Cursor ausschalten und ihn mittels Software simulieren. Wir ersetzen ihn dadurch, daß wir die Stelle, auf dem er sich befindet, revers darstellen.

```
a 0140 jsr   $cfbf
a 0143 jsr   $cecd
a 0146 lda   $fb
a 0148 pha
a 0149 lda   #$00
a 014b sta   $fb
a 014d php
a 014e cli
a 014f jsr   $db11
a 0152 ldy   $ca
a 0154 lda   ($c8),y
a 0156 ldx   #$ff
a 0158 cpx   $ff0d
a 015b bne   $0162
a 015d cpx   $ff0c
a 0160 beq   $016f
a 0162 stx   $ff0c
a 0165 stx   $ff0d
a 0168 stx   $0182
a 016b eor   #$80
a 016d sta   ($c8),y
a 016f ldx   $ef
a 0171 beq   $017f
a 0173 bit   $0182
a 0176 bpl   $017f
a 0178 eor   #$80
a 017a sta   ($c8),y
a 017c lsr   $0182
a 017f jmp   $ce54
a 0182 brk
a 0183 sei
a 0184 lda   #$40
a 0186 sta   $0312
a 0189 lda   #$01
a 018b sta   $0313
a 018e cli
a 018f rts
```

MINI-EDITOR

```
10 rem mini-editor=============c16 <cj>
20 rem (p) commodore welt team     <ho>
30 rem ========================== <ng>
40 rem (c) by alfons mittelmeyer   <cg>
50 rem                             <pd>
60 rem c16/116/p4                  <bg>
70 rem ========================== <po>
80 fori=1630to1716                 <pl>
90 reada:pokei,a:next              <dk>
100 data 169,000,133,008,032,232   <jo>
110 data 235,176,007,201,032,208   <bp>
120 data 002,169,160,024,096,164   <dg>
130 data 008,240,019,032,121,004   <ho>
140 data 201,160,208,003,032,115   <hn>
150 data 004,032,165,004,200,201   <jm>
160 data 000,208,248,096,076,086   <il>
170 data 137,072,165,153,201,003   <ha>
180 data 240,027,173,180,006,201   <ip>
190 data 013,240,005,104,072,032   <oo>
200 data 075,236,104,201,013,240   <hb>
210 data 004,201,032,208,003,141   <gb>
220 data 180,006,024,144,004,104   <lg>
230 data 032,075,236,169,252,133   <mn>
240 data 015,096,013,772,111,006   <oe>
250 data 802,094,006,804,137,006   <jg>
260 fori=1to3:reada:readb:readc    <he>
270 pokea,b:pokea+1,c:next:new     <ik>
280 rem ========================= <lo>
290 rem  p r o g r a m m e n d e  <no>
300 rem ========================= <kn>
```

VERSTELLEN DES INTERRUPTVEKTORS

Der Interruptvektor darf nicht durch einen Poke verbogen werden. Der Interrupt könnte nämlich genau zwischen dem Poken von Low- und High-Byte auftreten, auf einen nur teilweise verstellten Vektor treffen und ins Leere stürzen. Ein Systemabsturz wäre die Folge. Die Maschinenroutine von $0183 bis $018f unterbindet durch SEI den Interrupt. Der Vektor ITIME kann dadurch problemlos auf den neuen Wert eingestellt werden.

REAKTION AUF TASTENDRUCK

Den Anfang der Interruptroutine im ROM von $CE42 bis $CE53 haben wir in das RAM von $0140 bis $0151 kopiert. Der letzte Befehl lautet somit JSR $DB11. Dies ist die Routine SCNKEY, die für die Tastaturabfrage verantwortlich ist.
Sie bringt die gedrückte Taste in den Tastenpuffer oder aktiviert den Funktionstastenpuffer, falls eine Funktionstaste gedrückt ist.
Daß uns kein Tastendruck entgeht ist wichtig, damit wir den simulierten Cursor rechtzeitig ausschalten können. Wenn der Rechner, von uns unbemerkt, eine Taste zu lesen bekäme, würde er den Originalcursor auf die neue Position stellen. Der simulierte reverse Cursor würde einfach stehenbleiben, wenn es sich dabei um eine Cursortaste oder die Returntaste handeln würde.

HARD- UND SOFTWARECURSOR

Zum Simulieren des Cursors brauchen wir den Ort, an dem er sich befindet. In Adresse $CA findet sich die Spaltenposition, in den Zeilen $C8 und $C9 die Zeilenanfangsadresse. Mit dem Programmteil von $0152 bis $0155 bekommen wir die Spaltenposition in das Y-Register und das eventuell zu invertierende Zeichen in den Akku. Anschließend erfolgt die Abfrage, ob der Hardwarecursor eingeschaltet ist. Dies ist der Fall, wenn die Speicherstellen $FF0C und FF0D nicht beide den Wert $FF aufweisen.
In Adresse $0156 bis $0161 erfolgt die Abfrage und die entsprechende Verzweigung. Ist der Cursor eingeschaltet, schalten wir ihn aus. Die Speicherstelle $0182 dient als Flag, mit dem wir uns den Cursorzustand merken. Dabei soll ein Wert von $80 bis $FF einen eingeschalteten Cursor bedeuten und ein Wert von $00 bis $7F einen ausgeschalteten. Das Zeichen an der Cursorposition wird invertiert zurückgeschrieben.

ABSCHALTEN DES SOFTWARECURSORS

Auf die Routine von $0162 bis $016E, die den Hardwarecursor aus- und den Softwarecursor einschaltet, folgt zu guter Letzt die Abschaltung des Softwarecursors. Wenn eine Taste gedrückt wurde, ist die Abschaltung vorzunehmen.

NDX, der Tastaturindex in Speicherstelle $EF, informiert uns über die Anzahl der im Tastaturpuffer erfaßten gedrückten Tasten. Ist der Wert Null, so erfolgt der Rücksprung in die Original-Interrupt-Routine auf $CE54. Im anderen Fall ist zu prüfen, ob der Softwarecursor eingeschaltet ist. Unser Flag, die Speicherstelle $0182, gibt darüber Auskunft.
Durch eine weitere Invertierung sorgen wir wieder für den ursprünglichen Zustand des Zeichens an der Cursorposition. Das Flag läßt sich durch Rechtsverschieben oder Incrementieren mit einem einzigen Befehl auf den Wert $7F oder $00 bringen, was beides einen ausgeschalteten Softwarecursor bedeutet. Unter dem Namen BLINKWEG liegt dieses Programm auch als BASIC-Listing vor.

NICHTBLINKENDER CURSOR

```
10 rem blinkweg================c16 <ep>
20 rem (p) commodore welt team      <ho>
30 rem =========================== <ng>
40 rem (c) by alfons mittelmeyer    <cg>
50 rem                              <pd>
60 rem c16/116/plus4                <kb>
70 rem =========================== <po>
80 fori=320to399                    <ja>
90 reada:pokei,a:next               <dk>
100 data 032,191,207,032,205,206    <hp>
110 data 165,251,072,169,000,133    <hi>
120 data 251,008,088,032,017,219    <mi>
130 data 164,202,177,200,162,255    <ki>
140 data 236,013,255,208,005,236    <bi>
150 data 012,255,240,013,142,012    <il>
160 data 255,142,013,255,142,130    <hh>
170 data 001,073,128,145,200,166    <jm>
180 data 239,240,012,044,130,001    <fj>
190 data 016,007,073,128,145,200    <dn>
200 data 078,130,001,076,084,206    <ab>
210 data 000,120,169,064,141,018    <mg>
220 data 003,169,001,141,019,003    <hh>
230 data 088,096                    <ib>
240 sys387:new                      <mg>
250 rem ========================== <jf>
260 rem  p r o g r a m m e n d e   <bn>
270 rem ========================== <ld>
```

SYSTEMUHR

Eine Uhr in BASIC zu programmieren, ist weiter nicht schwierig. Zum Uhrzeit setzen und Uhrzeit lesen braucht nur die Variable TI$ gesetzt oder ausgegeben werden. Ein solches BASIC-Programm hat jedoch einen entscheidenden Nachteil: Während das Uhrprogramm läuft, können wir mit dem Rechner nichts anderes mehr tun. Wenn wir das Uhrprogramm verlassen, sehen wir die Zeit nicht mehr.
Eine Uhr, die ständig die Zeit anzeigt, läßt sich in Maschinensprache durch Systemeinbindung programmieren. Also nichts wie ran, eine Routine schreiben, die TI$ ausgibt, und sie einbinden. Dies ist aber leichter gesagt als getan. TI$ existiert in BASIC nur formal als Variable.

A$=TI$

Hier wird der Variablen A$ die Uhrzeit als Textstring übergeben. TI$ selbst ist jedoch keine Stringvariable, deren Inhalt an einer Stelle im Hauptspeicher zu finden ist, sondern eine Funktion wie zum Beispiel CHR$ oder LEFT$. Zwar findet sich die Uhrzeit in Form von sechzigstel Sekunden in den Speicherstellen $A3, $A4 und $A5. Das Umrechnen in das Stringformat ist aber sehr aufwendig. Daher wählen wir ein einfacheres Verfahren, die eigene Programmierung einer Uhr.
Was wir zu einer Digitaluhr benötigen, sind ein Taktsignal und drei Zähler für Sekunden, Minuten und Stunden. Das Taktsignal liefert der Systeminterrupt ITIME, der alle fünfzigstel Sekunden auftritt. Beim Interrupt ist nur ein Zähler hoch- oder runterzuzählen; nach fünfzig Interrupts drehen wir die Uhr eine Sekunde weiter und geben die Zeit aus.

Beim Weiterdrehen der Uhr werden zuerst die Sekunden um eins erhöht. Bei 60 Sekunden sind die Sekunden auf Null zu setzen und die Minuten zu erhöhen, bei 60 Minuten die Stunden. Bei 24 Uhr setzen wir die Stunden wieder auf Null.
Folgende Speicherstellen verwenden wir für unser Programm:

$D0: Stunden
$D1: Minuten
$D2: Sekunden
$D3: Bildschirmcode für die Ziffer Null
$D4: Bildschirmcode für Doppelpunkt
$D5: Low-Byte Bildschirmadresse
$D6: High-Byte Bildschirmadresse
$D7: Taktzähler

Vor dem Start des Maschinenprogrammes sind diese Speicherzellen richtig zu initialisieren. Stunden, Minuten und Sekunden werden im BCD-Format erfaßt, damit die Umrechnung in ASCII-Ziffern erleichtert wird. Als Bildschirmcode für die Ziffer Null kommen entweder $30, dezimal 48, oder $B0, dezimal 176, in Frage, je nachdem, ob normale oder inverse Ausgabe gewünscht wird. Für den Doppelpunkt sind es die Werte $3A, dezimal 58, oder $BA, dezimal 186. Wer ein anderes Trennzeichen haben möchte, kann einen anderen Wert eingeben.
Den Ort der Uhr auf dem Bildschirm geben $D5 und $D6 an. Aus Zeile und Spalte läßt sich die Adresse mit AD=3072 +40*Z+S errechnen. Der Taktzähler, der bei jedem Interrupt erniedrigt wird, sollte zu Beginn auf $32, dezimal 50, gesetzt werden, damit genau eine Sekunde später die richtige Zeit ausgegeben wird.
Um die Übersichtlichkeit zu erhöhen, wurde das Assemblerlisting durch dazwischengeschobene Bemerkungen in Abschnitte unterteilt. Falls Sie es mit dem Monitor eingeben möchten, sind die Bemerkungen wegzulassen, da der Monitor mit diesen nichts anzufangen weiß. Da zu diesem Programm auch ein BASIC-Listing existiert, ist die Eingabe mit dem Monitor unnötig.

SYSTEMUHR
Assemblermnemonics

Abfrage

```
a 065e dec $d7
a 0660 bne $068b
```

Hauptroutine

```
a 0662 lda #$32
a 0664 sta $d7
a 0666 ldx #$02
a 0668 jsr $068e
a 066b dex
a 066c bpl $0668
a 066e lda $d0
a 0670 cmp #$24
a 0672 bne $0677
a 0674 inx
a 0675 stx $d0
---------------
a 0677 ldx #$02
```

```
a 0679 ldy  #$07
a 067b jsr  $06a3
a 067e dex
a 067f bpl  $067b
a 0681 lda  $d4
a 0683 ldy  #$02
a 0685 sta  ($d5),y
a 0687 ldy  #$05
a 0689 sta  ($d5),y
```

ROM-Interrupt

```
a 068b jmp  $ce42
```

Uhr weiterstellen

```
a 068e lda  $d0,x
a 0690 sed
a 0691 clc
a 0692 adc  #$01
a 0694 cld
a 0695 sta  $d0,x
a 0697 cmp  #$60
a 0699 beq  $069e
a 069b ldx  #$00
a 069d rts
a 069e lda  #$00
a 06a0 sta  $d0,x
a 06a2 rts
```

Zeit ausgeben

```
a 06a3 lda  $d0,x
a 06a5 and  #$0f
a 06a7 ora  $d3
a 06a9 sta  ($d5),y
a 06ab lda  $d0,x
a 06ad lsr
a 06ae lsr
a 06af lsr
a 06bf lsr
a 06b1 and  #$0f
a 06b3 ora  $d3
a 06b5 dey
a 06b6 sta  ($d5),y
a 06b8 dey
a 06b9 dey
a 06ba rts
```

Systemeinbindung

```
a 06bb sei
a 06bc lda  #$5e
a 06be sta  $0312
a 06c1 lda  #$06
a 06c3 sta  $0313
a 06c6 cli
a 06c7 rts
```

Ende

DER INTERRUPT-ZÄHLER

Zu Beginn wird der Zähler $D7 erniedrigt. Wenn er noch nicht den Nullstand erreicht hat, erfolgt der Sprung zum normalen ROM-Interrupt. Andernfalls wird zuvor die Uhrroutine abgearbeitet. Der Zähler $D7 ist wieder auf $32, dezimal 50, zu setzen.

WEITERDREHEN DER UHR

In einer Schleife von X=2 bis X=0 werden Sekunden, Minuten und Stunden weitergezählt. Die Bearbeitung der jeweiligen Speicherstelle $D2 bis $D0 besorgt die Unterroutine von $068E bis $06A2. Sie erhöht die betreffende Zahl durch BCD-Addition mit der Zahl Eins. Ist der neue Wert kleiner als $60, so wird der Schleifenindex X auf Null gesetzt. Die Schleife in der Hauptroutine wird dadurch vorzeitig verlassen, so daß nicht auch noch die Minuten und Stunden erhöht werden.
Ist der neue Wert dagegen $60, so wird er gegen $00 ausgetauscht. Da nichts am Indexregister gedreht wird, bearbeitet die Schleife in der Hauptroutine auch die folgenden Stellen der Uhrzeit.
Den Stunden wird eine zusätzliche Behandlung zuteil. Nicht erst bei $60 sind sie auf Null zu stellen, sondern bereits bei $24. Durch Verwendung von INX braucht es nur ein einziges Byte, um eine Null in das X-Register zu bekommen.

AUSGABE DER UHRZEIT

Nach dem Weiterdrehen der Uhr soll die Zeit auch ausgegeben werden. Das X-Register dient wiederum als Index für unsere Sekunden, Minuten und Stunden.
Das Y-Register steuert die Positionierung der auszugebenden Ziffern. Wie eben läuft die Ausgabeschleife von X=2 bis X=0. Den jeweiligen Wert bringt die Routine von $06A3 bis $06BA auf den Bildschirm.
Zuerst wird aus der BCD-Zahl die niederwertige Ziffer durch Undieren mit $0F isoliert. Durch Odern mit dem Inhalt von $D3 wird der richtige Bildschirmcode erzeugt, der mit Hilfe des Zeigers $D5 und dem Y-Register auf den Bildschirm ausgegeben wird. Die höherwertige Ziffer wird durch viermaliges Schieben nach rechts gewonnen. Das Undieren mit $0F, das mit in das Listing gerutscht ist, ist überflüssig.
Erniedrigen des Y-Registers ist nötig, um die Ausgabe der nächsten BCD-Zahl vorzubereiten, die in den nächsten Schleifendurchläufen erfolgt. Danach ist nur noch das Trennzeichen aus der Speicherstelle $D4 in die Zwischenräume einzufügen und die Interruptroutine im ROM kann zum Abschluß aufgerufen werden.

PROGRAMM-BEDIENUNG

Das BASIC-Programm SYSTEMUHR fordert nach dem Starten zur Eingabe der Uhrzeit auf. Stunden, Minuten und Sekunden sind jeweils zweistellig und durch Schrägstriche getrennt einzugeben. Nach dem Drücken einer weiteren Taste wird die Uhrzeit in der linken oberen Bildschirmecke, revers dargestellt, sichtbar. Damit die Zeitausgabe beim Arbeiten nicht zu sehr verwirrt, empfiehlt es sich, mit ESC und T am Anfang der zweiten Bildschirmzeile ein Window einzurichten.
Weil nicht immer eine reverse Darstellung oder die Positionierung in der linken oberen Bildschirmecke wünschenswert ist, hat das Listing SYSTEMUHR einen Anpassungsteil. Es können Zeilen von null bis 24 und Spalten von null bis 32 gewählt werden. Die Uhr bleibt auch bei Bildschirmlöschen, Nachladen von Programmen und sonstigen Operationen erhalten.
Das Starten von reinen BASIC-Programmen macht der Uhr nichts aus. Programme, die Maschinensprache-Teile und Pokes enthalten, können jedoch einen Absturz herbeiführen. Selbst in SCRIPT/PLUS oder der eingebauten Plus4-Software haben Sie stets die Uhrzeit vor sich. Da Datenübertragungen über den Kassettenport oder den seriellen Port den Interrupt unterbinden, tritt dort ein vorübergehender Stillstand der Uhr ein, der zum Nachgehen führt. Bei der Floppy 1551 ließ sich ein solcher Effekt nicht feststellen.

NACHTRÄGLICHES STELLEN DER UHR

Sollte ein Nachstellen der Uhr erforderlich sein, nachdem das BASIC-Programm SYSTEMUHR nicht mehr im Hauptspeicher des Rechners vorhanden ist, kann dies mit dem Maschinenmonitor bequem vollbracht werden. Geben Sie einfach ein:

MONITOR
M D0

Die ersten drei Werte rechts neben dem Datum geben die Uhrzeit wieder. Wäre es zum Beispiel 12 Uhr 15 und 20 Sekunden, so wäre zu lesen:

>00D0 12 15 20 . . .

Durch Überschreiben erfolgt die neue Einstellung. Soll auf Sekundenbruchteile richtig eingestellt werden, so ist der letzte Wert in dieser Zeile auf $32 zu setzen.
Auch die Darstellung, ob normal oder revers sowie die Positionierung können auf die gleiche Weise geändert werden. Wurde mit RUN/STOP und der Reset-Taste die Uhr außer Gefecht gesetzt, so schaltet sich in BASIC

SYS1723 die Uhr wieder ein.
Bei einem vollständigen Reset sind vorher noch die acht Bytes von Adresse $D0 bis $D7 wieder auf die richtigen Werte zu bringen. Bei reverser Darstellung in der linken oberen Bildschirmecke und Uhrzeit wie vorher angenommen, wäre dies:

```
>00D0 12 15 20 B0 BA
      00 0C 32
```

MODIFIKATIONEN

Möchten Sie die Uhr schneller oder langsamer laufen lassen, so braucht nur der Lade-Befehl an Adresse $0662 abgeändert werden. Das Weiterzählen der Uhr kann in zeitlichen Abständen von einem bis zu 256 fünfzigstel Sekunden vorgenommen werden.

```
A 0662 LDA #$01;
          1/50 Sek.
A 0662 LDA #$FF;
        256/50 Sek.
```

In manchen Programmen werden wir mit langen Wartezeiten konfrontiert. Ob das Programm in einer Endlosschleife hängt oder wie lange wir noch zu warten haben, darüber kann uns auch eine rückwärts bis Null zählende Uhr informieren. Den Count-Down-Betrieb erreichen wir durch:

```
A 0691 SEC
A 0692 SBC  #$01
A 0697 CMP  #$99
A 069E LDA  #$59
```

Dazu sollte auch das Abschalten der Uhr bei Erreichen des Nullstandes ergänzt werden.

```
A 0670 CMP   #$59
A 0674 BEQ   $06C8
A 06C8 LDA   #$42
A 06CA STA   $0312
A 06CD LDA   #$CE
A 06CF STA   $0313
A 0602 BNE   $068B
```

Wer die Uhr gerne weiter ausbauen möchte, kann sich einmal an einer Weckfunktion versuchen. Gutes Timing brauchen nicht nur Uhren, sondern es ist auch wünschenswert bei manchen Bewegungsabläufen in Computerspielen. Der Interrupt sorgt für die unabhängige Steuerung mehrerer Objekte und wird von diesen nicht in seinem konstant wiederkehrenden Ablauf beeinflußt. Vielleicht kommen Ihnen hierzu einige Ideen in den Sinn.
Wir hoffen, daß wir Ihnen das Verfahren der Systemeinbindung etwas näher bringen und einige Anregungen geben konnten. *a.m.* □

DATENTRANSFER

Mit der Buschtrommel begann es

Von jeher war der Mensch auf Informationen und Nachrichtenaustausch erpicht. Dabei spielte natürlich die Geschwindigkeit des Austausches eine große Rolle. Zuerst wurden Kuriere losgeschickt, später dann Rauch- und Feuerzeichen verwendet. Die Urwaldtrommler könnte man als erste akustische Nachrichtenübermittler bezeichnen. Als das technische Zeitalter einsetzte, wurde der Informationsaustausch immer schneller und bequemer. Durch die Erfindung des Telefons gelang es, Informationen in großer Menge zu übermitteln. In unserer heutigen Zeit übernehmen diese Aufgaben immer mehr die Computer, zuerst im professionellen und zunehmend im privaten Bereich.

Es gibt viele Möglichkeiten, Texte von einem Computer zum anderen zu schicken. Den einfachsten Weg bietet die Telekommunikation.
Benötigt werden eine Telefonanlage, für jeden Computer je ein Akustikkoppler, das abgedruckte Programm für Ihren Rechner und ein Terminal-Programm für die Gegenstelle.
Bei einem Telefongespräch wird aus dem Schalldruck des gesprochenen Wortes eine Spannung mit wechselnder Frequenz erzeugt. Diese wird über die Leitung übertragen und Leitungsverluste durch Verstärker wieder ausgeglichen. Beim Empfänger wird diese Wechselspannung wieder in einen hörbaren Schalldruck zurückverwandelt.
Digitale Signale, die bei unserer Datenfernübertragung vorkommen, müssen in akustische Signale umgewandelt werden. Nur dann können sie auch über die Leitung gesendet werden. Beim Empfänger angekommen, werden diese akustischen Signale wieder in digitale umgesetzt. Die Geräte, die eine solche Umsetzung bewerkstelligen, nennt man Modems oder Akustikkoppler.
Die Übertragungs-Geschwindigkeit beträgt 300 Baud im Vollduplexverfahren. Das heißt, es werden in einem Datenkanal gleichzeitig Daten gesendet und empfangen. Dieses Verfahren wurde in Europa genormt und trägt die Bezeichnung CCITT (Comite Consultative International Telegraphique et Telefonique).
Es gibt zwei verschiedene CCITT-Normen. Zum einen die CCITT-V.21-Norm, die zwei gleich große Datenkanäle besitzt. Damit lassen sich aber nur Geschwindigkeiten von 300 Baud oder weniger erreichen.
Wollen Sie aber mit höheren Übertragungsarten arbeiten, so können die Daten nicht mehr gleichzeitig empfangen und gesendet werden. Das liegt daran, daß in einem Telefonkanal keine zwei Datenkanäle mit sicherem Störabstand untergebracht werden können. Sie müssen also auf das Halb-Duplexverfahren, die CCITT-V.23-Norm, umschalten.
Hierbei werden die Daten entweder gesendet oder empfangen. Das Ende einer Übertragung wird als Schlußzeichen oder über einen langsamen Datenkanal übermittelt. Im Hauptkanal werden Daten mit 1200 bps (Bits per Second) übertragen, ein Zusatzkanal mit 75 bps wird für die Steuerung der Datenübertragung verwendet.

DIE RS-232-SCHNITTSTELLE

Den Datenfluß zwischen Computer und anderen Peripherie-Geräten wie Drucker oder Akustikkoppler regelt die Schnittstelle RS-232, in Europa auch V24 genannt. Sie ist am geeignetsten für unsere Übertragung per Telefon. Die Übertragung erfolgt immer seriell, ein paralleler Transfer über Telefon ist nicht möglich. Bei der seriellen Übertragung werden die acht Bits eines Bytes einzeln übertragen. Dadurch wird zwar die Geschwindigkeit geringer, aber Sie kommen mit weniger Leitungen aus.
Die Schnittstelle am Commodore ist durch ein

Modul erhältlich, beziehungsweise ein Kabel, das auf der Commodore-Seite einen Userportstekker und auf der anderen einen 25poligen Stecker (DB25) aufweist. Diese Kabel gibt es in der Regel zu kaufen, sie nehmen auch die erforderliche Pegelumwandlung ±12 Volt vor. Sollten Sie aber vorhaben, dieses Kabel selbst zu basteln, so entnehmen Sie bitte die Pin-Belegung der beiden Stekker unserer Grafik.
Achten Sie aber bei der Verwendung des Userports als Eingang unbedingt auf die Eingangsspannung. Sie darf nur in einem Bereich von null bis fünf Volt liegen.
Soll der Userport als Ausgang dienen, halten die Ausgänge nur die Belastung eines TTL-Einganges aus. Es empfiehlt sich in jedem Fall, eine Pufferstufe einzubauen.

OHNE TERMINAL-PROGRAMM GEHT NICHTS

Das nächste wichtige Handwerkszeug zur Übertragung von Daten ist das Terminalprogramm. Wollen Sie zwei Commodore-Rechner verbinden, reicht das Grundprogramm des zweiten Listings.
Unser Beispiel zeigt Ihnen ein kleines Terminalprogramm für die Verbindung zwischen Commodore und PC. In Zeile 100 wird die RS-232-Schnittstelle geöffnet, der Bildschirm gelöscht und auf Kleinschrift umgeschaltet. Den OPEN-Befehl sind Sie von der Floppy gewohnt. Die Geräteadresse der RS-232-Schnittstelle lautet immer 2. Danach folgt die Parametereinstellung (Baud, Stopbits, Parität) mittels zweier CHR$-Anweisungen. In unserem Beispiel reicht für eine Übertragung mit 300 Baud, acht Bits, Vollduplex und keiner Paritätsprüfung der Befehl Chr$(6)+Chr$(5). Für eine Übertragung mit 300 Baud, sieben Bits, Vollduplex und keiner Parität müßte der Befehl OPEN2,2,0,Chr$(38) +Chr$(229) lauten.
Der Commodore legt sich nach dem OPEN-Befehl zwei Puffer mit je 256 Bytes an. Im einen werden die empfangenen und im anderen die zu sendenden Daten bis zur Weiterverarbeitung aufbewahrt. Die Geschwindigkeit, mit der die Daten an die Schnittstelle geschickt werden, ist immer konstant. Daher ist es bei einer größeren Geschwindigkeit nötig, die Daten in einem Puffer zwischenzulagern.
Das gleiche geschieht mit jenen Daten, die bereits im Rechner angekommen sind, aber noch nicht weiterverarbeitet wurden. In Zeile 110 werden dann zwei Variablen zur Datenkonvertierung initialisiert. G% steht für die Empfangsdaten und H% für die Sendedaten.
Da bei allen Commodore-Heimcomputern der ASCII-Code umgekehrt zum Standard-ASCII-Code verschlüsselt ist, muß eine Codewandlung vorgenommen werden.

Dies geschieht in den Zeilen 130 bis 170. Die Codewandlung betrifft aber nur die Klein- und Großbuchstaben. Bei Satzzeichen und ähnlichem entspricht der Commodore-Code dem des Standards.
Wird eine Taste gedrückt und liegt der ASCII-Wert des eingelesenen Zeichens im Bereich zwischen 64 und 91, dann ist es ein Kleinbuchstabe. Soll dieses Zeichen gesendet werden, so muß der Wert 32 addiert werden (Zeile 140). Das DELETE-Zeichen (20) wird in Zeile 160 zum Standard-BACKSPACE-Zeichen (8) umgewandelt.
Beim Empfang eines Zeichens wird vom ASCII-Wert 32 abgezogen, wenn es ein Kleinbuchstabe war (Zeile 150).
Vergleichen Sie dazu die

TERMINAL-PROGRAMME

```
10 rem terminal1================p4 <jo>
20 rem (p) commodore welt team      <ho>
30 rem ========================== <ng>
40 rem (c) by andy greil           <bo>
50 rem                             <pd>
60 rem uebertragung 8 bit          <no>
70 rem von commodore-rechner       <bh>
80 rem auf fremdrechner            <ai>
90 rem ========================== <jg>
100 open 2,2,0,chr$(6)+chr$(5):pri
ntchr$(12)+chr$(14):printchr$(147) <lc>
110 dim g%(256),h%(256)            <nm>
120 for s=0to255:h%(s)=s:next      <dh>
130 fors=32to64:h%(s)=s:next       <mk>
140 fors=65to90:h%(s)=s+32:next    <ad>
150 fors=97to122:h%(s)=s-32:next   <hk>
160 h%(20)=8                       <pf>
170 fors=193 to218:l=s-128:h%(s)=l
:next                              <gd>
180 fors=0to255                    <mi>
190 l=h%(s)                        <mg>
200 ifl<>0theng%(l)=s              <pm>
210 next                           <ch>
220 print"Uebertragung-Start"      <ie>
230 get#2,a$                       <oa>
240 ifa$=""orst<>0then270          <nf>
250 ifa$=chr$(10)then230           <fh>
260 print chr$(g%(asc(a$)));       <ge>
270 geta$                          <kf>
280 ifa$=chr$(95)then310           <ja>
290 ifa$<>""thenprint#2,chr$(h%(as
c(a$)));                           <im>
300 goto 230                       <hm>
310 print:print"Filename:";:poke19
,1:inputb$                         <id>
319 poke19,0:print:open8,8,8,b$+",
s,r"                               <nk>
320 get#8,a$:su=st:print#2,a$;:pri
nta$;                              <id>
330 ifa$<>""thenprint#2,chr$(h%(as
c(a$)));                           <od>
340 ifsu=0then320                  <ad>
350 close8:goto230                 <nn>
360 rem ========================== <bn>
370 rem  p r o g r a m m e n d e   <aa>
380 rem ========================== <da>
```

TERMINAL-PROGRAMME

```
10 rem terminal2================p4 <ik>
20 rem (p) commodore welt team      <ho>
30 rem ========================== <ng>
40 rem (c) by andy greil           <bo>
50 rem                             <pd>
60 rem uebertragung 7 bit          <ng>
```

Tabelle der Standard-Codes mit der aus Ihrem Handbuch.

VON DER THEORIE ZUR PRAXIS

Mit unserem kleinen Terminalprogramm, das Sie abgetippt und auch hoffentlich abgespeichert haben, können Sie bereits mit Mailboxen oder anderen Computern kommunizieren.
Als erstes verbinden Sie Ihren Rechner mit dem Akustikkoppler, schalten ihn ein und stellen den Wahlschalter auf ORIG (originate). Sollten Sie nicht mit einer Mailbox, sondern mit einem anderen Computer kommunizieren wollen, so muß dessen Akustikkoppler auf ANSW (answer) geschaltet sein.
Schalten Sie Ihren Computer ein und starten Sie das abgetippte Terminalprogramm.
Plazieren Sie Ihr Telefon so, daß Sie den Telefonhörer bequem mit dem Koppler verbinden können. Achten Sie unbedingt auf die richtige Lage des Hörers. Als nächstes wählen Sie die Nummer des gewünschten „Ansprechpartners". Ertönt nach dem Rufzeichen ein hoher Pfeifton, ist die Gegenstelle zum Datenaustausch bereit. Wenn Sie mit einer Mailbox verbunden sind, so bestätigen Sie mit der RETURN-Taste, daß auch Sie bereit zum Datenaustausch sind. Der weitere Verlauf ergibt sich durch das Mailboxmenü. Da alle Mailboxen mit unterschiedlicher Menü-Führung arbeiten, gibt es keine einheitlichen Wegweiser durch die Systeme. Sie müssen sich selbst einarbeiten.
Wenn Sie ein vorbereitetes sequentielles File senden wollen, so drücken Sie die „Pfeil-nach-links-Taste". Daraufhin werden Sie nach dem Filenamen gefragt. Geben Sie den Namen Ihres vorbereiteten Files an und bestätigen Sie mit RETURN. Ihr File wird nun zur Gegenstelle gesandt.
Unser kleines Terminalprogramm ist natürlich nicht der Weisheit letzter Schluß, es soll Ihnen vielmehr die Grundzüge der Datenfernübertragung demonstrieren und Sie zum weiteren Ausbau des Programmes inspirieren.

```
70 rem zwischen cbm-rechnern                   <fh>
80 rem                                         <cp>
90 rem ============================            <jg>
100 open 2,2,0,chr$(38)+chr$(229)              <nm>
110 printchr$(147);chr$(14)                    <cb>
130 geta$: ifa$=""then170                      <mc>
135 printa$;                                   <mn>
140 a=asc(a$):ifa<91anda>64thena=a
+32                                            <hb>
150 ifa=20 thena=8                             <oe>
160 print#2,chr$(a);                           <dn>
170 get#2,a$:ifa$=""then130                    <pj>
175 ifa$=chr$(10)then170                       <lm>
180 a=asc(a$):b=a:ifa>96thenb=a-32             <pi>
185 ifa<91 and a>64 then b=a+128               <bk>
190 ifa=8then b=20                             <op>
200 a=b:print chr$(a);:goto130                 <pb>
210 rem ========================               <ai>
220 rem  p r o g r a m m e n d e               <fg>
230 rem ========================               <kk>
```

ASCII-STANDARD-CODE

Char Hex Dec Char Hex

Char	Hex	Dec	Char	Hex	Dec
A	41	65	a	61	97
B	42	66	b	62	98
C	43	67	c	63	99
D	44	68	d	64	100
E	45	69	e	65	101
F	46	70	f	66	102
G	47	71	g	67	103
H	48	72	h	68	104
I	49	73	i	69	105
J	4A	74	j	6A	106
K	4B	75	k	6B	107
L	4C	76	l	6C	108
M	4D	77	m	6D	109
N	4E	78	n	6E	110
O	4F	79	o	6F	111
P	50	80	p	70	112
Q	51	81	q	71	113
R	52	82	r	72	114
S	53	83	s	73	115
T	54	84	t	74	116
U	55	85	u	75	117
V	56	86	v	76	118
W	57	87	w	77	119
X	58	88	x	78	120
Y	59	89	y	79	121
Z	5A	90	z	7A	122

AKUSTIKKOPPLER ODER MODEM?

Im Grunde sind auch die Akustikkoppler Modems. Denn das Wort Modem ist die Abkürzung von Modulator-Demodulator. Das, was wir im Hausgebrauch Modem nennen, sind Modems, die galvanisch direkt mit der Telefonleitung verbunden sind.
Im Gegensatz dazu gibt es die Akustikkoppler, die Signale akustisch empfangen und senden.
Bei der Übertragung mit Akustikkopplern gibt es verschiedene Fehlerquellen. Die Probleme beginnen schon bei der Plazierung des Telefonhörers in die Muffen des Akustikkopplers: Meistens ist bei den neuen Apparaten der Post der Hörer zu groß, bei den alten dagegen zu klein. Das hat zur Folge, daß die akustischen Signale, die gesendet und empfangen werden, nicht richtig umgesetzt werden. Die Übertragung ist somit gefährdet, denn schon der kleinste Huster veranlaßt den Akustikkoppler, wirre Zeichen auf den Bildschirm zu bringen und auch zu übertragen.
Abhilfe bringen da die Modems. Da sie direkt an der Telefonleitung angeschlossen sind, fallen die Probleme mit störenden Geräuschen in der Umgebung weg. Gekoppelt mit einem guten Terminalprogramm, können Sie sogar Ihr Modem Telefonnummern wählen lassen und das so lange, bis ein Anschluß erreicht ist. Ihr Computer nimmt Ihnen

sogar die lästige „Ein-log-Prozedur" beim Anschluß mit Mailboxen ab. Auch können Sie damit Anrufe von anderen Computerbesitzern empfangen, die eine eigene Mailbox betreiben.
Leider gibt es für die Commodore-Homecomputer wie C128, C64 oder Plus/4 noch keine von der Deutschen Bundespost zugelassenen und mit einer ZZF-Prüfnummer versehenen Modems. Es werden zwar einige von verschiedenen Firmen angeboten, aber diese dürfen Sie nur an Haustelefonanlagen und Nebenstellen betreiben.
Also bleibt nur der gute alte Akustikkoppler und das Warten, bis sich unsere liebe Post endlich entschließt, nicht nur an die professionellen PC-User, sondern auch an die Homecomputer-Besitzer zu denken.

Computer lassen sich aber nicht nur per Akustikkoppler oder Modem miteinander verbinden. Die direkte Verbindung ist immer noch die einfachste Lösung. Diese Art wird auch Null-Modem genannt.
Für die Datenübertragung (Received Data, Transmitted Data, Signal Ground) sieht die RS-232-Norm nicht nur Kabel für die Datenübertragung vor, sondern auch solche für die Steuerung der Kommunikation. Da die Request-to-send (RTS)- und die Celar-to-Send (CTS)-Leitungen nicht eindeutig definiert sind, können die Anschlüsse bei verschiedenen Rechnern unterschiedlich ausfallen.
Verbinden Sie alle Leitungen zwischen den beiden Computern gekreuzt. Also die RTS (request to send) mit der CTS (clear to send)-Leitung und umgekehrt. Die Signal-Erde wird mit der Signal-Erde verbunden. Die Leitungen TXD (Transmitted Data) und RXD (Received Data) werden ebenfalls kreuzweise verbunden. Funktioniert die Übertragung nicht, so versuchen Sie, die Leitungen ungekreuzt miteinander zu verbinden. Das wäre RTS mit RTS und CTS mit CTS. Sollte sich immer noch kein Erfolg einstellen, so verbinden Sie auch noch die Leitung TXD und RXD direkt. Zum besseren Verständnis betrachten Sie bitte unsere Grafik. □

SO WÄRE ES BESSER GEWESEN

Das Listing fehlte

C16-P4-SPECIAL Nr. 3 *Seite 36*
S&A BASIC

In dem Artikel wurde das Beispiellisting S&A-BASIC-DEMO angesprochen, das aber nicht mit abgedruckt war. Deshalb wird es hier nachgeliefert.

```
S&A BASIC DEMO
10 rem s&a basic demo==========c16 <fj>
20 rem (c) by michael inden         <dg>
30 :#ztrans120:#chn120:poke65287,1
36                                  <ei>
40 printchr$(147)"    michael inde
n presents s&a basic    "           <of>
50 fort=0to79:#scroll0,0,0,39,0,0:
:forw=0to75:nextw,t:scnclr          <al>
60 #zdesign1,52,24,52,24,24,52,24,
52                                  <le>
70 #zdesign2,255,24,52,24,24,52,24
,52                                 <ik>
80 #zdesign3,52,24,52,24,24,52,24,
255                                 <mf>
90 #zdesign4,52,24,52,24,24,52,24,
52                                  <pe>
100 :#fill0,0,29,12,65              <gl>
110 fort=37to0step-1                <pp>
120 forx=8to0step-1                 <ij>
130 :#setsprite1,t,5,x,0            <hc>
140 forw=0to40:next                 <ik>
150 :#spriteoff1                    <ea>
160 #animate1,1:#animate1,1         <fn>
170 next:next                       <ff>
180 scnclr                          <fb>
190 print"aaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaa"                    <ik>
200 print"bbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbb"                    <oa>
210 print"ccccccccccccccccccccccc
ccccccccccccccc"                    <gd>
220 print"ddddddddddddddddddddddd
ddddddddddddddd"                    <eh>
230 #animate1,0:#animate2,2:#anima
te3,3:#animate4,1:goto230           <gh>
240 rem =p=r=o=g=r=a=m=m=e=n=d=e==  <ak>
```


IMPRESSUM C16/P4

erscheint zweimonatlich in der CA-Verlags GmbH, einer Gesellschaft der Aktuell-Gruppe

VERANTWORTLICH FÜR DEN INHALT:
Anton Kult
Alfons Mittelmeyer

REDAKTION UND STÄNDIGE MITARBEITER:
Peter Basch, Harald Beiler, Rosemarie Huber, Lothar Miedel, Michael Reppisch, Rudolf Schmid-Fabian, Torsten Seibt, Hermann Wellesen, Bernd Welte

GESCHÄFTSFÜHRER:
Werner E. Seibt

ANSCHRIFT FÜR ALLE VERANTWORTLICHEN:
Postfach 1161,
8044 Unterschleißheim
Tel.: 089/1298011
Telex: 5214428 cav-d

ANZEIGENVERWALTUNG:
ADV-Mediendienste,
Aindlingerstr. 17–19,
8900 Augsburg 1

Tel.: 08121/7904-227
Telex: 533502
Teletex: 821887
Telefax: 0821/7904-243

VERANTWORTLICH FÜR DEN ANZEIGENINHALT:
Brigitte Kostić
Es gilt Preisliste Nr. 9 vom 1.6.1988
Media-Unterlagen bitte anfordern.

VERTRIEB:
Verlagsunion Wiesbaden

Printed in Germany

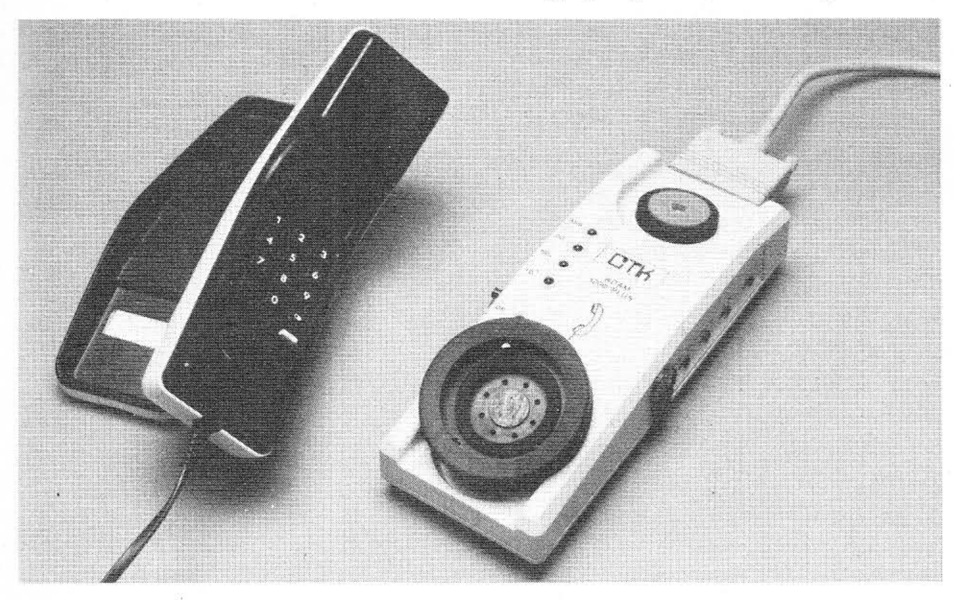
DFÜ-LEXIKON

Von A bis Z

In vielen Büchern über Datenfernübertragung finden sich unverständliche Fachausdrücke und Abkürzungen. Wir wollen Ihnen einen kleinen Überblick der häufigsten Bezeichnungen geben.

ANSWER
Senden mit dem Antwortfrequenzpaar f1=1850 Hz für 0
f2=1650 Hz für 1.

ASCII
American Standard Code of Information Interchange.
ASCII ist eine genormte Code-Wandlung, die jedem Buchstaben, jeder Zahl und Sonderzeichen einen bestimmten numerischen Wert zuordnet. Dieser Code wird von den meisten Computern verstanden und wird auch als Standard im DFÜ-Bereich verwendet.



BAUD
Gibt die Übertragungsgeschwindigkeit an. Anzahl der Signalwechsel pro Sekunde.

BPS
Eine andere Möglichkeit, die Übertragungsgeschwindigkeit zu bezeichnen.
Anzahl der übertragenen Bits pro Sekunde.

BTX
Bildschirmtext der Post; Übertragung mit 1200/75 Baud.

CARRIER
Carrier bedeutet auf deutsch „Träger“ und bezeichnet im allgemeinen den Ton, mit dem Computer signalisieren, daß sie zur Kommunikation bereit sind. Dieses Carrier-Signal ist ein gleichbleibender Pfeifton mit einer bestimmten Frequenz. Mailboxen zum Beispiel warten bei Anruf, ob ein solcher Carrier vorhanden ist, erst dann beginnen sie mit dem Senden von Daten.

CBBS
Computer Bulletin Board System.

CCITT
Comite Consultative International Telegraphique et Telefonique.
Europäische Norm für Telekommunikation.

CTS
Clear To Send oder empfangsbereit.
Leitung, die anzeigt, daß ein DCE-Gerät bereit ist, Daten vom DTE-Gerät zu empfangen. Diese Leitung ist im Normalfall AUS. Wenn über die RTS-Leitung Sendebereitschaft angezeigt wird, so löst das DCE-Gerät mit CTS EIN das Senden von Daten aus. Ist die RTS-Leitung auf AUS geschaltet, so wird auch die CTS-Leitung auf AUS geschaltet (siehe auch RTS).

CUG
Closed User Group, spezielle Teilnehmergruppe im Datex-P-Netz.

DATEX-P
Datennetz der Deutschen Bundespost.
Übertragungsgeschwindigkeit bis 9600 Baud.

DCD
Empfangspegel vorhanden.
Leitung, über die ein DCE-Gerät einem DTE-Gerät anzeigt, daß es ein Signal empfängt. Dies veranlaßt das DTE-Gerät, ein RTS-Signal zu senden.

DCE
Data Communication Equipment.
Gerät, mit dem Daten empfangen werden können (Modem, Drucker und so weiter).

DSR
Datenempfänger betriebsbereit.
Leitung, die dem DTE-Gerät anzeigt, daß das DCE-Gerät betriebsbereit ist. Im Normalfall ist diese Leitung auf EIN geschaltet.

DTE
Data Terminal Equipment.
Gerät, das Daten sendet, zum Beispiel Terminal, Computer.

DTR
Data Terminal Ready. Das DTE-Gerät zeigt mit einem EIN-Signal an, daß es betriebsbereit ist.

DOWN-LOAD
Bezeichnung für das Einlesen von Daten (Texte, Programme) aus einem anderen Rechner.

FSK
Frequency Shift Keying.
Daten werden durch Frequenzumtastung moduliert.

ISDN
Integrated Services Digital Network.
Sämtliche Daten, auch Sprache, werden digital übertragen.

LOGIN
Verbindungsaufbau zu einem Rechner, meistens mit einer User-ID.

LOGOFF
Gegenteil von LOGIN.
Wird meistens mit den Befehlen LOGOFF oder BYE durchgeführt.

MAILBOXEN
Mailboxen nennt man in unserer heutigen Zeit Computer, die über normalen Telefonanschluß oder DATEX-P angerufen werden können. Mailboxen stellen dem Anrufer

sogenannte „Bretter" zur Verfügung, auf denen öffentliche Mitteilungen an andere Anrufer geschrieben oder deren Nachrichten abgerufen werden können. Um einen persönlichen „Briefkasten" in einer Mailbox zu erhalten, ist es in der Regel notwendig, sich als „User" eintragen zu lassen. In diesen Briefkasten können andere Teilnehmer persönliche Nachrichten schreiben, die nur der eingetragene User lesen kann. Außerdem können Sie Mailboxen als elektronische Tageszeitungen, produziert von den Anrufern, betrachten.

MODEM
Modulator-Demodulator.
1. Das Galvanische Modem:
Es wird direkt an der Telefonleitung angeschlossen. Muß von der Bundespost genehmigt und angeschlossen werden.
2. Der Akustik-Koppler:
Der Telefonhörer wird auf Gummimanschetten des Kopplers aufgesteckt. Keine direkte elektrische Verbindung zum Telefonnetz.
Nur mit ZZF-Nummer der Bundespost genehmigt.

NUA
Network User Adress.
Teilnehmerkennung im DATEX-P-Netz. Gibt die anzurufende Nummer an.

NUI
Network User Identification.
Teilnehmerkennung im DATEX-P-Netz. Gibt die eigene Nummer an. Dient zur Abrechnung der empfangenen Daten durch die Bundespost und kann nur dort beantragt werden.

ORIGINATE
Senden mit dem Sendefrequenzpaar f1=1189 Hz für 0
f2= 980 Hz für 1 (siehe ANSWER).

PAD
Packet Assembly and Dissassembly Facility.
DATEX-P-Vermittlungsstelle.
Übernimmt die Umwandlung des asynchronen Datenflusses in genormte „Pakete" und umgekehrt.

PARAMETER
Als Parameter werden sämtliche Einstellungen im Übertragungsprotokoll bezeichnet (siehe auch PROTOKOLL).

PARITÄT
1. Gerade Parität:
Das Paritäts-Bit wird gesetzt, wenn die Anzahl der Datenbits im gesendeten Zeichen eine gerade Zahl ergibt.
2. Ungerade Parität:
Ist im gesendeten Zeichen die Anzahl der Datenbits ungerade, wird das Paritätsbit gesetzt.
3. Keine Parität:
Bei der Übertragung wird auf das Senden des Paritätsbits verzichtet.

PROTOKOLL
1. Festlegung der Regeln bei der Datenübertragung (zum Beispiel Baud, Übertragungsart, Parität).
2. Aufzeichnung der empfangenen Daten.

RS-232
Genormte Schnittstelle zur seriellen Übertragung von Daten.

RTS
Ready To Send oder Request To Send (sendebereit).
Leitung, mit der vom DTE-Gerät beim DCE-Gerät angefragt wird, ob Daten gesendet werden können. Im Normalfall ist diese Leitung AUS. Sollen Daten gesendet werden, so wird vom DTE-Gerät ein EIN-Signal an das DCE-Gerät gesendet. Die Übertragung beginnt, wenn das DCE-Gerät über die CTS-Leitung ein EIN-Signal sendet. Beim Beginn der Übertragung wird das RTS-Signal auf AUS gesetzt.

RxD
Recieved Data.
Über diese Leitung werden Daten vom DTE-Gerät zum DCE-Gerät gesendet.

SG
Signal Ground.
Erdungsleitung.

STARTBIT
Zeigt den Beginn eines zu sendenden Zeichens an.

STOPBIT(S)
Kennzeichnung des Übertragungsendes eines Zeichens. Meist werden ein oder zwei Stopbits verwendet.

SysOp
System Operator.
Bezeichnung für den Betreiber einer Mailbox.

TERMINAL
Im allgemeinen wird damit das Programm bezeichnet, das der Computer braucht, um mit anderen Rechnern zu kommunizieren.
Außerdem werden auch die einzelnen Arbeitsplätze einer Großrechenanlage als Terminal bezeichnet.

TxD
Transmitted Data.
Über diese Leitung werden die Daten vom DTE-Gerät zum DCE-Gerät übermittelt.

UP-LOAD
Bezeichnung für das Senden von Daten (Texte, Programme) an einen anderen Rechner.

USER-ID
User Identification, Benutzererkennung.
Nach Eingabe der User-ID erkennt das angerufene System, ob der Anrufer zugriffsberechtigt ist oder nicht.

ÜBERTRAGUNGSARTEN

1. SIMPLEX:
Übertragung nur in eine festgelegte Richtung, vom Sender zum Empfänger, möglich (Rechner–Drucker).

2. HALBDUPLEX:
Datenübertragung in beide Richtungen, aber immer nur abwechselnd. Wird bei sehr hohen Übertragungsgeschwindigkeiten benutzt.

3. VOLLDUPLEX:
Die am meisten benutzte Übertragungsart.
Datenübertragung in beide Richtungen gleichzeitig.

V.21
Protokoll für eine Übertragung mit 300 Baud.

V.24
Europäische Norm der RS-232-Schnittstelle.

X.25
CCITT-Protokoll für synchrone Datenübertragung.

X-MODEM
In vielen neueren Terminalprogrammen für Computer befindet sich der Menüpunkt X-MODEM. Damit lassen sich Programme sofort lauffähig übertragen, ohne den lästigen Umweg über ASCII-Codewandlung.

XON/XOFF PROTOKOLL
Vereinbarte Steuerung des Datenflusses während der Datenübertragung.
XON – (CTRL-Q):
Signal für die Gegenstation zum Senden der Daten.
XOFF – (CTRL-S):
Signal für die Gegenstation, daß sie das Senden von Daten unterbrechen soll.

G-und S-Shape

Shapes bieten erstaunliche Möglichkeiten. Trotz der dürftigen Beschreibung im Rechnerhandbuch brauchen Sie darauf nicht zu verzichten. Wir füllen die Informationslücke.

Vor allem das Abspeichern und Einladen der als Strings definierten Shapes macht vielen Hobby-Programmierern Probleme. Will man sich eine SHAPE-Bibliothek anlegen, so ist es sehr umständlich, die SHAPE-Daten immer in Datazeilen abzulegen. Schließlich liegen die Werte bereits als Strings vor und es erscheint zunächst sehr einfach, diese Strings mit PRINT#8,A$ abzuspeichern und mit INPUT#8, A$ wieder einzuladen. Das Abspeichern erfolgt noch völlig problemlos, wie das Beispiel ab Zeile 1000 in unserem DEMO-Programm (Listing 1) zeigt. Der Zusatz ",s,w" öffnet ein sequentielles File zum Schreiben (englisch: write). Bei Kassettenbetrieb beachten Sie die entsprechenden Hinweise im Handbuch zum OPEN-Befehl (mit Sekundäradresse 2).
Wollen Sie diesen SHAPE-String jedoch einfach mit INPUT# einlesen, tauchen die Probleme auf. Der INPUT-Befehl erkennt nämlich einige Zeichen als Trennungszeichen, zum Beispiel die CHR$-Codes 13 (RETURN), 44 (Komma) und 58 (Doppelpunkt). Steht eines dieser Zeichen im SHAPE, so wird der Rest des SHAPEs nicht eingelesen.
Ein weiterer Nachteil des INPUT#-Befehls ist die Tatsache, daß damit das File-Ende nicht abgefragt werden kann, denn die Variable ST wird erst dann ungleich Null, wenn die EOF-Marke (End of File) gefunden wird. Dann bleibt das Programm hängen, denn es kommt kein Zeichen mehr, das dem INPUT#-Befehl das Ende des Strings anzeigt.
Viele Programmierer umgehen diese Schwierigkeiten, indem sie den String Zeichen für Zeichen in ASCII-Werte umwandeln und diese als Zahlen abspeichern und einlesen. Der entscheidende Nachteil hierbei ist, daß dafür wesentlich mehr Speicherplatz und daher auch mehr Einlesezeit benötigt wird. Es ist wesentlich besser, auf den GET#-Befehl zurückzugreifen.

STRINGS SCHEIBCHENWEISE EINLESEN

Zunächst muß das SEQ-File mit dem Zusatz ",s,r" für Lesen (englisch: read) geöffnet werden (OPEN1,1,0,"name" bei Kassette). Jetzt muß nur noch ein Zeichen nach dem anderen mit GET#8,A$ eingelesen werden, wie dies ab Zeile 1030 gezeigt wird. Aber Vorsicht! Auch hier sind einige Fallstricke vorhanden. Benutzen Sie die Routine, wie sie abgedruckt ist. Vor allem drei Fehlerquellen gibt es:
Fehler eins: Wird Q$ direkt zu R$ addiert, so werden Nullen, die in den meisten SHAPEs zahlreich vorhanden sind, als Leerzeichen interpretiert, und bei der Addition mit R$=R$+Q$ bleibt R$ unverändert (was normalerweise auch vernünftig ist). Dagegen wird R$ tatsächlich um einen „Nullstring" verlängert, wenn im Programm R$=R$ +CHR$(0) steht.
Fehler zwei: Beim Umwandeln von Q$ in den ASCII-Wert Q muß ein CHR$(0) zu Q$ addiert werden, da ein Leerstring im ASC-Befehl sonst ei-

```
10 rem shape-demo===============p4 <ka>
20 rem (p) commodore welt team      <ho>
30 rem ========================== <ng>
40 rem (c) by r.schmid-fabian       <jp>
50 rem                              <pd>
60 rem                              <ah>
70 rem basic v3.5                   <nl>
80 rem plus4 (c16/116 + 64 kb)      <fd>
90 rem ========================== <jg>
100 trap320                         <fm>
110 rem ****** shapes lesen ****** <fh>
120 fora=0to123:readc:a$=a$+chr$(c
):next:gosub530                     <ho>
130 fora=0to123:readc:b$=b$+chr$(c
):next:gosub530                     <ah>
140 rem ** shapes untereinander ** <jf>
150 graphic1,1:gshapea$:gshapeb$:g
osub550                             <ph>
160 rem ** shapes nebeneinander ** <bm>
170 x0=100:y0=100                   <hb>
180 la=len(a$):ba=asc(mid$(a$,la-3
,1)):rem breite von shape a$        <bp>
190 ha=asc(mid$(a$,la-1,1)):rem ho
ehe von shape a$                    <gg>
200 gshapea$,x0,y0,1:gshapeb$,x0+b
a,y0,1:gosub550                     <mo>
210 rem *** shapes verkleinern *** <ie>
220 mid$(a$,la-3,1)=chr$(ba-7):rem
 rechte seite um 7 bit kuerzen      <lh>
230 mid$(b$,la-3,1)=chr$(ba-7)      <nk>
240 graphic1,1:gshapea$,x0,y0,1:gs
hapeb$,x0+ba-7,y0,1:gosub550        <ga>
250 rem ** shapes uebereinander ** <on>
260 graphic1,1:forx0=10to280step42
:gshapea$,x0,y0,4:gshapeb$,x0+ba-1
8,y0-5,4:next                       <mi>
270 rem * shape stueckchenweise ** <fg>
280 gosub550:graphic1,1             <ho>
290 fori=1to120:c$=c$+chr$(0):next
:c$=c$+chr$(39)+chr$(0)+chr$(23)+c
hr$(0)                              <pd>
300 restore430:fora=1to120:readc:m
id$(c$,a,1)=chr$(c):gshapec$,100,1
00:next                             <ca>
310 gosub550                        <el>
320 graphic0:end                    <lg>
330 rem ****** shape 1 *********** <jj>
340 data0,0,48,0,0,0,0,72,112,0,0,
0,132,112,0,0                       <bn>
350 data1,2,112,0,0,2,49,112,0,0,4
,72,240,0,0,8                       <cb>
360 data72,112,0,0,16,48,48,0,0,32
,0,16,0,0,64,0                      <hn>
370 data8,0,0,128,0,4,0,1,0,0,2,0,
3,255,255,255                       <gh>
380 data0,2,0,0,1,0,2,252,1,253,0,
2,132,1,5,0                         <gg>
390 data2,132,1,5,0,2,252,249,253,
```

```
0,2,0,136,1,0,2                      <kd>
400 data0,136,1,0,2,0,152,1,0,2,0,
136,1,0,2,0                          <dm>
410 data136,1,0,3,255,255,255,0,39
,0,23,0                              <gi>
420 rem ****** shape 2 *********** <nn>
430 data0,1,252,0,0,0,3,254,0,0,0   <cg>
440 data7,255,0,0,0,15,255,128,0    <db>
450 data0,31,255,192,0,0,63,255,22
4,0,0,63,255,224,0,0                 <jf>
460 data63,255,224,0,0,63,255,224,
0,0,63,255,224,0,0,63                <fo>
470 data255,224,0,0,31,255,192,0,0
,15,255,128,0,0,7,255                <al>
480 data0,0,0,3,254,0,0,0,1,252,0,
0,0,0,112,0                          <cg>
490 data0,0,0,112,0,0,0,0,112,0,0,
0,0,112,0,0                          <cg>
500 data0,0,112,0,0,0,0,112,0,0,0,
0,112,0,0,0                          <ej>
510 data0,112,0,0,39,0,23,0          <cf>
520 rem **datazeile zeigen ******* <bi>
530 print,peek(63)+256*peek(64):re
turn                                 <fi>
540 rem **** warten auf taste ****
*                                    <jh>
550 char,10,23,"        ":char,10,23
,"taste":getkeyq$:return             <bj>
1000 rem *** shape abspeichern *** <mn>
1010 graphic0:open8,8,8,"shape a,s
,w":print#8,a$:close8                <ch>
1020 rem *** shape einladen ****** <oa>
1030 open8,8,8,"shape a,s,r"         <dj>
1040 get#8,q$:q=asc(q$+chr$(0)):if
st=0thenr$=r$+chr$(q):goto1040       <ma>
1050 close8                          <pb>
1060 graphic1,1:gshaper$:getkeya$:
graphic0                             <ip>
1070 rem ========================= <jo>
1080 rem fuer c16/116 ohne           <pm>
1090 rem erweiterung alle rem-       <dn>
1100 rem zeilen und die programm-    <bg>
1110 rem zeilen ab 1000 loeschen     <hp>
1120 rem ========================= <cd>
```

```
10 rem shapeeditor============c16  <lb>
20 rem (p) commodore welt team      <ho>
30 rem ========================== <ng>
40 rem (c) r. schmid-fabian         <fe>
50 rem c16/116 plus4                <oe>
60 rem ========================== <nk>
70 trap260                          <ho>
80 scnclr:char ,5,5,"bitte shape m
it * zeichnen                       <ec>
90 char,1,7,"untere rechte ecke mi
t # kennzeichnen"                   <ie>
100 char,8,9,"danach erst return"  <ob>
110 ti$="000000":wait164,1:rem pau
se                                  <ed>
120 rem tastatur als eingabekanal* <me>
130 scnclr:printchr$(27)"m";:rem s
croll aus                           <ok>
140 open1,0:input#1,q$:close1       <hj>
150 printchr$(27)"l":rem scroll ei
n                                   <ao>
160 rem * shape zeichnen ********* <gf>
170 ba=3072:dim x,y,xm,ym:x0=140:y
0=90                                <ip>
180 graphic1,1:g=35:s=32            <id>
190 for y=0 to 24:for x=0 to 39     <kh>
200 w=peek(ba):if w=g then 240      <ll>
210 ba=ba+1                         <mg>
220 if w<>s then draw 1,x0+x,y0+y   <di>
230 nextx:nexty                     <pk>
240 sshape sh$,x0,y0,x0+x-1,y0+y-1 <dk>
250 getkeyq$                        <fp>
260 printchr$(27)"l":graphic0       <ip>
270 scnclr:input"SHAPE als DATA-Ze
ilen ausgeben";q$:if q$="n" then e
nd                                  <pi>
280 rem * shape sh$ in data ****** <fd>
290 zn=1000:a=1:e=len(sh$):z=2:pri
ntchr$(147);                        <nc>
300 printzn"dA";e;":rem start"      <cf>
310 zn=zn+10:printzn"dA ";          <od>
320 for a=a to a+15                 <lo>
330 if a>e then printchr$(157)" ":
print"renU10,10":goto360            <kc>
340 printmid$(str$(asc(mid$(sh$,a,
1))),2)",";:next                    <ea>
350 printchr$(157)" ":z=z+1:goto31
0                                   <cj>
360 tp=1319:poketp,19:fori=1toz:po
ketp+i,13:next:poke239,i:end        <bj>
370 rem ========================== <ie>
380 rem  ab hier in eigenes         <gd>
390 rem  programm uebernehmen       <np>
400 rem ========================== <ic>
410 read d:sh$=""                   <km>
420 for i=1 to d:read a:sh$=sh$+ch
r$(a):next                          <fa>
430 rem =e=n=d=e================== <ld>
```

```
10 rem =file to data =======c16/p4 <jm>
20 input"prg-name";p$               <ej>
30 open8,8,8,p$:",s,r"              <ib>
40 zn=1000:fora=0to123:input#8,c    <pd>
50 a$=a$+mid$(str$(c),2)+",":iflen
(a$)>25thenprintzn;"data "+a$:a$="
":zn=zn+10                          <ma>
60 next:printzn;"data "+a$+chr$(20
):close8                            <kc>
70 rem =e=n=d=e=================== <ml>
```

nen SYNTAX ERROR erzeugt. Der Interpreter unterscheidet also zwischen Leerzeichen mit der Länge 0 und CHR$(0) mit der Länge 1.
Fehler drei: Das Ende des Strings muß richtig erfaßt werden. Der PRINT#-Befehl schreibt nämlich am Ende des Strings ein RETURN-Zeichen (CHR$ (13)). Dieses wird natürlich durch GET# mit erfaßt und an R$ angefügt. Das hat zur Folge, daß das HI-Byte der SHAPE-Länge in Y-Richtung 13 ist (normalerweise null) und der Computer versucht, 13*256 Zeilen darzustellen.
Davon ist er leider auch mit der STOP-Taste nicht abzuhalten. Daher darf die String-Addition erst erfolgen, nachdem entschieden ist, ob das Fileende schon gefunden wurde (ST > 0).
Werden mehrere SHAPEs in ein einziges File geschrieben, funktioniert diese Methode natürlich nicht. Der einfachste Weg ist dann, als erste abgespeicherte Zahl die Länge des SHAPE-Strings abzuspeichern. Diese Länge wird dann auch als erstes eingelesen und bestimmt die Anzahl der Durchläufe in der For/Next-Schleife.

SHAPE ZU SHAPE GESELLT SICH GERN

Da ein String nur maximal 255 Zeichen umfaßt, ist die Größe eines SHAPEs sehr begrenzt. Sollen größere Minigrafiken dargestellt werden, müssen diese aus mehreren SHAPEs zusammengesetzt werden. Die einfachste Methode, zwei SHAPEs zusammenzufügen, ist, sie untereinander oder nebeneinander zu positionieren.
Dazu wird keinerlei Berechnung benötigt. Wie Zeile 150 im Demo-Programm zeigt, werden bei dem SHAPE-Kommando ohne x-y-Angabe die SHAPEs automatisch untereinander plaziert.
Steht hinter dem String keine Zahl, so wird als Startpunkt der Pixel-Cursor genommen. Dieser steht nach dem ersten SHAPE-Kommando an der linken unteren Ecke des SHAPEs.
Etwas schwieriger ist es, zwei SHAPEs direkt nebeneinander darzustellen. Ist die x-Ausdehnung des ersten SHAPEs bekannt, so braucht diese nur zu dessen Startwert addiert zu werden, um den Startwert des zweiten SHAPEs zu erhalten.
Eine allgemeinere Routine, die beliebige SHAPEs nebeneinander darstellt, muß die Breite des ersten SHAPEs ermitteln können. Dies geschieht in Zeile 180. Die letzten vier Bytes des SHAPEs enthalten in LO/HI-Darstellung die x- und die y-Ausdehnung. Hier wird nur das LO-Byte verwertet, da SHAPEs mit einer Breite größer 255 selten vorkommen. In Zeile 190 ist das analoge Verfahren zum Ermitteln der Höhe dargestellt.

ZU GROSSE SHAPES VERKLEINERN

Wird ein SHAPE mit Hilfe eines SHAPE-Editors erzeugt, so wird oft unnötigerweise ein viel größerer SHAPE abgespeichert, als gewünscht ist. Der ganze Editierbereich, meist eine Bildschirmseite mit 40*25 Zeichen, wird in ein SHAPE umgewandelt. Da die Darstellung eines SHAPEs relativ langsam erfolgt und stark von der Größe abhängt, sollten zu große SHAPEs verkleinert werden.
Eine Möglichkeit ist durch den GSHAPE gegeben. Liegt der zu große SHAPE als A$ vor, wird er mit GRAPHIC2,1: GHAPEA$ in der linken oberen Ecke dargestellt. Dann definieren wir uns mit SSHAPE B$,x0,y0, x1,y1 einen neuen, kleineren SHAPE. Er wird am besten mit GSHAPE B$, 100,100,1 zur Kontrolle invers dargestellt, bevor er, wie in Zeile 1010 gezeigt wurde, abgespeichert wird. Mit etwas Probieren sind die optimalen Werte für x0 bis y1 schnell gefunden.
Wie sich auf einfache Weise ein SHAPE verkleinern läßt, der nur rechts zu breit ist, wird in Zeile 220 gezeigt. Hierbei wird lediglich das LO-Byte des x-Wertes verändert.
Bei unserem SHAPE-EDITOR ist dies nicht nötig, da wir die Größe des SHAPEs durch Setzen eines #-Zeichens in die rechte untere Ecke selbst definieren können.
Die gesetzten Punkte werden durch „*" festgelegt, und nach RETURN wird das SHAPE in DATA-Zeilen umgewandelt.
Oft wird für ein Spiel die Grafik mit SHAPEs aufgebaut. Wenn die Bildteile nicht rechteckig sind, kann es vorkommen, daß das zweite SHAPE einen Teil des ersten überschreibt. In Zeile 260 wird gezeigt, wie dies mit der EOR-Verknüpfung vermieden wird.
Leider werden uns sehr selten Spiele zugesandt, die SHAPEs verwenden, obwohl dadurch der beste Hintergrund zu erzielen wäre. Der Grund liegt sicher in der Langsamkeit der SHAPE-Darstellung. Man kann jedoch die Vorteile eines veränderten Zeichensatzes und die Geschwindigkeit des CHAR-Befehls sehr gut mit der hervorragenden Grafikfähigkeit der SHAPEs verbinden. Das Hauptproblem dabei ist meist das Erkennen der Kollision der Spielfigur mit Hindernissen oder Schätzen. Versuchen Sie es doch einmal mit folgendem Tip:

Am Anfang wird ein Integerfeld mit DIM XY% (40,25) definiert; Integerfelder brauchen nur 2/5 des Platzes von normalen Feldern. In ihm werden die Spalten- und Zeilenpositionen, die das SHAPE jeweils abdeckt, als Eins gesetzt, wenn es ein Hindernis, und als Zwei, wenn es ein Schatz ist. Die Spielfigur wird mit verändertem Zeichensatz und dem CHAR-Befehl erzeugt und bewegt, wobei jede Bewegung mit dem XY%-Feld kontrolliert werden kann.

EIN TRICK FÜR SHAPE-SAMMLER

Wer aus einem fertigen Spiel ein SHAPE in seine Sammlung aufnehmen will, steht oft vor dem Problem, daß an den DATA-Zeilen der Anfang und das Ende eines bestimmten SHAPEs nicht erkennbar ist. Wenn davor noch ein langes Maschinenprogramm steht, wird das Auszählen sehr mühsam. Leichter erkennbar ist die FOR/NEXT-Schleife, mit der die Daten eingelesen werden.
Um die richtige Zeilennummer zu den entsprechenden Daten zu finden, gibt es einen einfachen Trick. Die Speicherstellen 63/64 enthalten das LO- beziehungsweise HI-Byte der aktuellen DATA-Zeile. Diese kann somit, wie in Zeile 530 des DEMO-Programms, berechnet werden.
Es gibt auch noch eine andere einfache Möglichkeit: Das Programm wird mit der STOP-Taste angehalten und das SHAPE als String wie in Zeile 1010 abgespeichert.
Mit Hilfe des kleinen Hilfsprogramms „File to Data" (Listing 2) können die Daten dann in DATA-Zeilen umgewandelt werden. Das Programm zeigt jedoch nur die Zeilen auf dem Schirm an, auf eine automatische Einbindung mit Hilfe der Tastaturpuffer-Methode wie im SHAPE-Editor wurde verzichtet. Meist reicht der Platz auf dem Schirm aus, um alle DATA-Zeilen darzustellen, die dann mit RETURN übernommen werden. □

HARDCOPIES
Grundlagen

GRAFIKBILDSCHIRM
Organisation und Hardcopies

Für jeden Computer und jeden Drucker

Hardcopy zu klein, zu groß, zu wenig dicht, falsches Verhältnis von Breite und Höhe oder für den speziellen Drucker nicht verfügbar? In zwei Folgen bringen wir Grundlagen, Möglichkeiten und Listings.

Mit den Grafikbefehlen des BASIC V3.5 kann jeder Bildpunkt auf bequeme Weise angesprochen werden. Schwierig aber wird es, wenn es gilt, eine Hardcopy des Grafikbildschirmes auszudrucken. Im Bedienerhandbuch zum Rechner ist darüber nichts zu finden, zumindest nicht im C16-Handbuch. Dieses bricht mitten im Anhang einfach ab.
Dem C116- oder Plus/4-Handbuch können wir auf Seite 228 entnehmen, daß der Bildschirmspeicher für die Grafik den Speicherplatz von $2000 bis $3FFF belegt. Um damit allerdings etwas anfangen zu können, bedarf es entweder weiterer Informationen oder einiger Experimente.
Wie wird die Grafik, die immerhin die Information über 64.000 Bildpunkte enthalten muß, in einem Adreßraum von etwa 8.000 Bytes gespeichert? Dies kann nur dadurch geschehen, daß ein Byte die Information über acht Bildpunkte enthält. Wie wir wissen, besteht ein Byte aus acht Bits. Jeder Bildpunkt kann entweder gesetzt oder nicht gesetzt sein. Ist er gesetzt, so ist auf dem Bildschirm die Vordergrundfarbe zu sehen, ansonsten die Hintergrundfarbe.
Gesetzte und nicht gesetzte Bildpunkte lassen sich in Form von gesetzten und nicht gesetzten Bits speichern. Wie dies im Einzelnen geschieht, wollen wir jetzt untersuchen. Dazu schalten wir die hochauflösende Grafik mit Text ein durch:

GRAPHIC 2,1

Außerdem ist die Anfangsadresse $2000 des Grafikbildschirmes in die dezimale Darstellung umzurechnen.

?DEC("2000")

Die dezimale Adresse lautet 8192. Ein kleiner Versuch könnte vielleicht etwas zeigen.

POKE 8192,255

Es erscheint ein kleiner waagrechter Strich in der linken oberen Ecke des Bildschirms. Die Zahl 255 wird in einem Byte durch acht gesetzte Bits realisiert. Acht nebeneinanderliegende schwarze Pünktchen, die im Computerjargon Pixel heißen, sind zu sehen. Ein gesetztes Bit bedeutet also ein gesetztes Pixel. Noch ist nicht klar, welches Bit welchem Pixel entspricht.

```
10 rem listing 1======(hardcopies) <nf>
20 rem by alfons mittelmeyer       <if>
30 rem ==========================  <ng>
40 graphic1,1:clr                  <ap>
50 ax=dec("2000")                  <fk>
60 fori=axtoax+7999                <cf>
70 pokei,255                       <mc>
80 char,0,24,hex$(i)+str$(i)       <aa>
90 getkeyx$                        <gd>
100 next                           <ek>
110 graphicclr                     <bb>
120 rem ========================== <gn>
130 rem  p r o g r a m m e n d e   <jf>
140 rem ========================== <ge>
```

```
10 rem listing 2======(hardcopies) <fc>
20 rem demoprogramm                <ao>
30 rem dieses programm liest ein   <ha>
40 rem buchstabenmuster, konver-   <np>
50 rem tiert es, und gibt es dabei <dk>
60 rem auf den bildschirm aus.     <dg>
70 rem --------------------------  <an>
80 rem  matrix aus zeichensatzrom  <ae>
90 rem --------------------------  <cf>
100 ax=53256:rem buchstabe a       <gd>
110 poke1177,62:rem peek aus rom   <fe>
120 fori=0to7                      <hp>
130 a(i)=peek(ax+i)                <ik>
140 next                           <jl>
150 poke1177,63:rem peek aus ram   <hi>
160 gosub200:goto160               <ed>
170 rem -------------------------- <ef>
180 rem  matrix-konvertierung      <kc>
190 rem -------------------------- <ca>
200 forj=0to7                      <ld>
210 a=0                            <co>
220 fori=7to0step-1                <ci>
230 a(i)=2*a(i)                    <dk>
240 a=a+a                          <en>
250 ifa(i)>255thena=a+1:a(i)=a(i)-
256:print"Q";:elseprint".";        <bk>
260 nexti:print                    <jg>
270 b(j)=a                         <cb>
280 nextj                          <ld>
290 fori=0to7                      <fk>
300 a(i)=b(i)                      <ha>
310 nexti:return                   <kl>
320 rem ========================   <il>
330 rem  p r o g r a m m e n d e   <al>
340 rem ========================   <oc>
```

POKE 8192,1
POKE 8192,128

Das linke Pixel wird durch das höchstwertige Bit vertreten, das rechte Pixel durch das niederstwertige Bit. Wir können gespannt sein, ob die nächste Speicheradresse die rechts oder unten anschließenden Pixel wiedergibt.

POKE 8192,128
POKE 8193,64
POKE 8194,32

```
listing 3===========(hardcopies)
hardcopy-maschinenteil
assemblerlisting
================================
. 0138   20 4e 01  jsr $014e
. 013b   a0 08     ldy #$08
. 013d   a2 08     ldx #$08
. 013f   16 d2     asl $d2,x
. 0141   2a        rol
. 0142   ca        dex
. 0143   d0 fa     bne $013f
. 0145   ea        nop
. 0146   ea        nop
. 0147   20 d2 ff  jsr $ffd2
. 014a   88        dey
. 014b   d0 f0     bne $013d
. 014d   60        rts
. 014e   a0 00     ldy #$00
. 0150   85 d2     sta $d2
. 0152   a2 08     ldx #$08
. 0154   b1 d0     lda ($d0),y
. 0156   95 d2     sta $d2,x
. 0158   05 d2     ora $d2
. 015a   85 d2     sta $d2
. 015c   e6 d0     inc $d0
. 015e   d0 02     bne $0162
. 0160   e6 d1     inc $d1
. 0162   ca        dex
. 0163   d0 ef     bne $0154
. 0165   60        rts
================================
```

```
10 rem listing 4=====(hardcopies)  <gk>
20 rem by alfons mittelmeyer       <if>
30 rem ==========================  <ng>
40 rem maschinensprachteil         <gn>
50 rem fuer hires-                 <ol>
60 rem hardcopyprogramme           <kg>
70 rem ==========================  <po>
80 fori=312to357                   <ig>
90 reada:pokei,a:next              <dk>
100 data 032,078,001,160,008,162   <mm>
110 data 008,022,210,042,202,208   <gd>
120 data 250,234,234,032,210,255   <ab>
130 data 136,208,240,096,160,000   <ej>
140 data 133,210,162,008,177,208   <dg>
150 data 149,210,005,210,133,210   <im>
160 data 230,208,208,002,230,209   <jg>
170 data 202,208,239,096           <dg>
180 rem ========================   <cn>
190 rem basicteil ist anzufuegen   <oi>
200 rem ========================   <ne>
```

POKE 8195,16
POKE 8196,8
POKE 8197,4
POKE 8198,2
POKE 8199,1

Die Reihenfolge vom höchstwertigen Bit zum niederstwertigen entspricht der Reihenfolge der Pixel von links nach rechts. Mit aufsteigenden Speicheradressen dagegen scheint es am Bildschirm abwärts zu gehen. Jedoch werden wir gleich eines Besseren belehrt.

POKE 8200,255

Wir haben damit die oberste Reihe der nächsten Spalte angesprochen. Da es wohl etwas mühsam ist, sich den ganzen Grafikbereich mit POKEs per Hand zu erschließen, soll ein kurzes BASIC-Listing namens HIRES uns die Arbeit abnehmen. Sie finden es als Listing Nr. 1 im Anhang zu diesem Artikel.

ACHT BYTES PRO ZEICHEN – 40 ZEICHEN PRO ZEILE

HIRES beschreibt 8.000 Adressen ab $2000 (8192 dezimal) mit dem Wert 255. Nach dem Programmstart ist in der linken oberen Bildschirmecke wieder der kurze Strich zu sehen, den wir schon von unseren POKE-Versuchen her kennen. Links unten am Bildschirm wird die jeweils beschriebene Speicheradresse in hexadezimaler und dezimaler Form dargestellt. Sobald irgendeine Taste gedrückt wird, ist die nächste Speicheradresse an der Reihe. Etwas Geduld ist vonnöten, soll die Organisation des Bildschirmaufbaus abgelesen werden. Nach achtmaligem Tastendruck befindet sich die kurze Linie in der zweiten Spalte. Wie weiteres Drücken zeigt, geht der Aufbau zeilenweise vonstatten. Acht Linien, die jeweils acht Pixel breit sind, füllen von oben nach unten den Raum, den im Textmodus ein einzelnes Zeichen auf dem Bildschirm benötigen würde.
Solche „Zeichen" werden zeilenweise aufgebaut, mit jeweils 40 „Zeichen" pro Zeile. Der Bildschirm hat, wie im Textmodus, auch jetzt 25 Zeilen, 40 mal acht Pixel ergeben 320 Bildpunkte in der Breite, 25 mal acht Pixel sind 200 Bildpunkte in der Länge. Wir wissen jetzt, wie die Bildpunkte der hochauflösenden Grafik untergebracht sind und können uns Gedanken über das Ausdrucken machen.

VORÜBERLEGUNGEN ZUM DRUCKEN

Statt eines einzelnen Byte für ein Zeichen haben wir im Grafimodus gleich acht Bytes vor uns. Im Textmodus hatten wir einen Code, der dem Drucker sagte, welches Zeichenmuster er aus seiner Bibliothek auswählen solle. Jetzt haben wir das Zeichenmuster selbst vorliegen.
Leider ist es anders organisiert, als es der Drucker verarbeiten kann. Auf dem Bildschirm bestimmt ein Byte acht nebeneinander liegende Pixel. Beim Drucker jedoch liegen die Nadeln nicht neben-, sondern untereinander. Würden wir die Zeichenmuster so an den Drucker senden, wie sie beim Rechner angeordnet sind, so erschienen sie auf dem Papier alle um 90 Grad gedreht. Vor dem Ausdruck ist folglich eine Konvertierung der Zeichenmatrix vorzunehmen. Listing Nr. 2 zeigt, wie das in BASIC realisiert werden könnte. Dieses Listing ist nur als Beispiel und nicht zum Abtippen gedacht.

MATRIX-KONVERTIERUNG

AX enthalte die Anfangsadresse des zu lesenden Zeichenmusters. Mit einer Schleife werden die Inhalte dieser und der folgenden sieben Adressen in die Variablen A(0) bis A(7) eingelesen. In einer weiteren Schleife mit J=0 bis J=7 soll das Zeichenmuster spaltenweise an den Drucker gesendet werden. Das an den Druk-

ker zu sendende Byte soll in Form der Variablen A bereitgestellt werden. Hierzu ist aus den Variablen A(0) bis A(7) jeweils das höchstwertige Bit zu entnehmen. Wir setzen die Variable A auf den Wert Null und sprechen durch eine weitere Schleife mit dem Index I=0 bis I=7 die indizierten Variablen A(0) bis A(7) an. Das höchstwertige Bit ist in die Variable A zu schieben. Hierzu schieben wir die Bits in den indizierten Variablen um eine Stelle nach links mit A(I)=2*A(I). Damit liegen die folgenden Bits für den nächsten Durchgang bereit. Das frühere Bit sechs ist somit jetzt zu Bit sieben geworden und kann im nächsten Durchgang mit erhöhtem J herausgeschoben werden.
Das erhaltene Bit ist jetzt in die Variable A hineinzuschieben. Da BASIC keine Schiebebefehle kennt, verschieben wir den Inhalt von A um eine Stelle nach links mit A=A+A. Hierzu ist jetzt das aus A(I) herausgeschobene Bit zu addieren. Wir realisieren dies durch die IF-Abfrage:

IFA(I)>255THENA=A +1:A(I)=A(I)−256

Der Übertrag von 256, der sich bei gesetztem höchstwertigen Bit ergab, mußte natürlich wieder abgezogen werden, so daß für die weitere Bearbeitung nur mehr die restlichen sieben Bits zur Verfügung stehen. Nach der Abarbeitung der inneren Schleife steht in A das an den Drucker zu sendende Byte bereit. Das höchstwertige Bit aus A(0) ist zum höchstwertigen Bit in A geworden, das höchstwertige Bit aus A (7) zum niederstwertigen Bit in A.

Viele Drucker sprechen mit dem höchstwertigen Bit die oberste Nadel an, manche aber mit dem untersten Bit. Für den letzteren Fall müßte die innere Schleife umgekehrt ablaufen, oder die Bits müßten in umgekehrter Richtung in die Variable A geschoben werden. Das erstere wäre zu verwirklichen durch die Schleife:

FOR I=7 TO 0 STEP−1

Das Schieben nach rechts geschähe durch:

A=A/2
IFA(I)>256THENA=A +128:A(I)=A(I)−256

Wenn wir davon ausgehen, daß vorher im Hauptprogramm der Druckerkanal durch den CMD-Befehl geöffnet worden ist, können wir mit PRINT CHR$(A) unser Byte auf den Drucker ausgeben. Nach acht Durchläufen sind alle acht Bytes ausgegeben und es kann der Rücksprung in das aufrufende Hauptprogramm mit RETURN erfolgen.

PROGRAMMIEREN IN MASCHINENSPRACHE

64 Schiebeoperationen braucht es, bevor ein ganzes Zeichen an den Drukker ausgegeben ist. In BASIC nimmt das eine Menge Zeit in Anspruch. Quälende Langsamkeit bei der Hardcopy wäre die Folge. Es ist daher zu überlegen, ob Listing Nr. 2 nicht besser durch ein Maschinenprogramm zu ersetzen wäre. Mit den Kenntnissen aus dem Einführungskurs im C16/P4-SPECIAL Nr. 3 ausgestattet, können wir uns getrost daranwagen. Damit das Maschinenprogramm nicht mit eventuellen Centronics-Routinen kollidiert, die der eine oder andere benötigt, um über Centronics-Kabel an nicht COMMODORE-kompatible Drukker zu gehen, verlegen wir die Maschinenroutine in den Stack-Bereich des Prozessors.

Ein Zeiger, mit dem die Adressen des Grafikbereichs indirekt ansprechbar sind, wird benötigt.

```
210 rem listing 5=====(hardcopies)  <oe>
220 rem by alfons mittelmeyer       <bi>
230 rem ==========================  <jc>
240 rem hires-hardcopy fuer         <fe>
250 rem grafikfaehige commodore-    <ah>
260 rem drucker.                    <ci>
270 rem nur in verbindung mit       <ia>
280 rem listing 4                   <ap>
290 rem ==========================  <bk>
300 poke208,0:poke209,32            <hb>
310 poke325,9:poke326,128:rem ora
#$80                                <ie>
320 poke321,106:rem ror             <dn>
330 open4,4:cmd4                    <ig>
340 printchr$(8);                   <ie>
350 forj=1to25                      <of>
360 print                           <gl>
370 fori=1to40:sys312:next:nextj    <bo>
380 print#4,chr$(15):close4         <gl>
390 rem ========================    <ag>
400 rem  p r o g r a m m e n d e    <hd>
410 rem ========================    <gk>
```

```
210 rem listing 6=====(hardcopies)  <cc>
220 rem by alfons mittelmeyer       <bi>
230 rem ==========================  <jc>
240 rem fuer epson-kompatible       <jm>
250 rem drucker                     <ag>
260 rem                             <jj>
270 rem nur in verbindung mit       <ia>
280 rem listing 4                   <ap>
290 rem ==========================  <bk>
300 poke208,0:poke209,32            <hb>
310 e$=chr$(27):x$=chr$(13)+chr$(1
0)                                  <ln>
320 open4,4:cmd4                    <hh>
330 printe$"a"chr$(8);              <pm>
340 forj=1to25                      <om>
350 printx$e$"k"chr$(64)chr$(1);    <kb>
360 fori=1to40:sys312:next:nextj    <dc>
370 print#4,e$"2"x$;:close4         <mi>
380 rem ==========================  <da>
390 rem  p r o g r a m m e n d e    <ko>
400 rem ==========================  <ic>
```

Dafür kommen nur die Adressen der Zero-Page in Frage. Die C16/116/P4-Rechner stellen die Adressen $D0 bis $E8 dem Benutzer zur freien Verfügung. $D0 und SD1 verwenden wir als Adreßzeiger.
Unter Umständen könnte es nützlich sein, ein Flag bereitzustellen, das dem BASIC-Programm meldet, ob ein Zeichen nicht gedruckt zu werden braucht, weil alle acht Bytes vielleicht nur den Wert Null aufweisen. Für dieses Flag wählen wir die Adresse $D2. Die Adressen $D3 bis SDA wollen wir für das zu lesende Zeichenmuster benützen. Das Programmieren kann beginnen.

LDY #$00
STY $D2

Das Flag wird auf den

```
210 rem listing 7=====(hardcopies) <ca>
220 rem by alfons mittelmeyer      <bi>
230 rem ========================== <jc>
240 rem fuer ibm-kompatible        <dk>
250 rem drucker                    <ag>
260 rem                            <jj>
270 rem nur in verbindung mit      <ia>
280 rem listing 4                  <ap>
290 rem ========================== <bk>
300 poke208,0:poke209,32           <hb>
310 e$=chr$(27):x$=chr$(13)+chr$(1
0)                                 <ln>
320 open4,4:cmd4:printe$":";       <lo>
330 printe$"a"chr$(8)e$"2"         <ja>
340 forj=1to25                     <om>
350 printx$e$"k"chr$(64)chr$(1);   <kb>
360 fori=1to40:sys312:next:nextj   <dc>
370 print#4,e$"a"chr$(12);         <lp>
380 print#4,e$"2"x$:close4         <me>
390 rem ========================== <ej>
400 rem  p r o g r a m m e n d e   <hd>
410 rem ========================== <bb>
```

```
210 rem listing 8=====(hardcopies) <cg>
220 rem by alfons mittelmeyer      <bi>
230 rem ========================== <jc>
240 rem universal-hardcopy         <be>
250 rem fuer alle drucker mit      <ci>
260 rem definierbarem geschaefts-  <ao>
270 rem zeichen. nur in verbindung <kf>
280 rem mit listing 4              <id>
290 rem ========================== <bk>
300 poke208,0:poke209,32           <hb>
310 x$=chr$(13):e$=chr$(8):f$=chr$
(15)                               <pe>
320 open5,4,5:open4,4              <md>
330 forj=1to25                     <nl>
340 print#4,e$x$f$;                <od>
350 fori=1to40:sys334:ifpeek(210)=
0then390                           <bh>
360 print#4,chr$(141);:poke2035,5:
sys65481                           <kd>
370 sys315:print#5:i$=str$(i-1)    <bm>
380 print#4,chr$(16)right$("0"+rig
ht$(i$,len(i$)-1),2)chr$(254);     <ln>
390 next:nextj                     <mc>
400 close4:close5                  <dl>
410 rem ========================== <bb>
420 rem  p r o g r a m m e n d e   <dj>
430 rem ========================== <im>
```

Wert Null gesetzt. Gleichzeitig soll das Y-Register zur indirekten Adressierung dienen und dabei den Wert Null aufweisen, da dabei keine Indizierung vonnöten ist.

LDX #$08

Das X-Register ist sowohl Zähler für unsere Leseschleife als auch Index zum Ablegen der gelesenen Bytes in die dafür reservierten Adressen $D3 bis $DA.

LDA ($D0),Y

STA $D2,X

Das durch unseren Zeiger in $D0 und $D1 adressierte Byte aus dem Grafikbereich wird in den Akku geladen und in eine durch das Register X indizierte Adresse von $DA bis $D3 geschrieben. Warum wir nicht mit $D3, sondern $DA beginnen, wird noch erklärt.

ORA$D2
STA $D2

Alle gelesenen Bytes werden geodert und das Resultat in $D2 abgelegt. Enthält $D2 am Ende immer noch den Wert Null, so bedeutet dies, daß jedes gelesene Byte ein Null-Byte war.

INC $D0
BNE $0162
INC $D1

Der Adreßzeiger wird um Eins erhöht. Wenn beim Erhöhen das Low-Byte den Wert Null annimmt, muß auch das High-Byte erhöht werden. Im anderen Fall wird der Befehl zum Erhöhen des High-Bytes übersprungen.

DEX
BNE $0154

Der Zähler und Index X wird um Eins verringert. Ist X anschließend noch nicht Null, erfolgt der Sprung zum Laden des nächsten Bytes mit LDA ($D0),Y. Wenn durch DEX das X-Register Null wird, so wird das Zero-Flag gesetzt, auf das der BNE-Befehl reagiert. Hätten wir X dagegen in umgekehrter Richtung von eins nach acht laufen lassen, wäre das nicht der Fall gewesen. Nach INX hätte erst ein zusätzlicher Befehl CPX #$09 eingeschoben werden müssen, den wir uns so erspart haben.

Statt X am Anfang mit LDX #$08 zu initialisieren, hätten wir genausogut LDX #$07 verwenden können. Das Speichern wäre dann eben mit STA $D3,X vonstatten gegangen und die Schleifenendabfrage nach DEX mit BPL.

RTS

GESONDERTE LESEROUTINE

Die Routine zum Lesen der Bytes aus dem Grafikbereich wurde als Unterprogramm angelegt, damit sie gegebenenfalls von BASIC aus gesondert angesprungen werden kann, um zu sehen, ob etwas auf den Drucker ausgegeben werden müßte. Wir kommen jetzt zur gesamten Routine, mit Matrixkonvertierung und Druckerausgabe.

JSR $014E

Am Anfang steht der Aufruf der Unterroutine zum Lesen der acht Bytes aus dem Grafikbereich.

LDY #$08
LDX #$08

An Stelle der Schleifenvariablen J und K nehmen wir in Maschinensprache die Register X und Y.

ASL $D2,X
ROL

Mit dem ASL-Befehl wird das höchstwertige Bit aus der durch X indizierten Zeile herausgeschoben. Genausogut hätte das auch ein ROL-Befehl getan. Der folgende ROL-Befehl schiebt dieses Bit von rechts nach links in den Akku. Bei Druckern, die ihr Byte andersherum brauchen, muß lediglich ROL gegen ROR ausgetauscht werden.

DEX
BNE $013F

Solange X noch nicht Null ist, wird auf den Befehl ASL $D2,X zurückgesprungen. Erst wenn

alle acht höchstwertigen Bits erfaßt sind, wird die Schleife verlassen.

NOP
NOP

Diese zwei NOP-Befehle bewirken nichts. Sie reservieren nur Platz. Commodore-Drucker brauchen die Bits in umgekehrter Reihenfolge, was ja leicht zu ändern ist. Außerdem drucken sie Grafik nicht mit acht, sondern nur mit sieben Nadeln. Bit sieben, das höchstwertige Bit, ist hierbei auf Eins zu setzen. Damit die Operation ORA #$80 in dieses Programm eingebaut werden kann, wurde der hierzu benötigte Platz mit Hilfe der beiden NOP-Befehle geschaffen.

JSR FFD2

Nachdem das Byte im Akku bereitsteht, kann es an den Drucker ausgegeben werden. Die Betriebssystemroutine BSOUT, die durch JSR $FFD2 angesprochen wird, gibt den Akkuinhalt auf das gewählte Ausgabegerät aus.

DEY
BNE $013D
RTS

Acht Bytes insgesamt sind auszugeben. Die Kontrolle darüber obliegt dem Y-Register. Ist dieses nach dem Erniedrigen noch nicht Null, erfolgt der Sprung wieder auf LDX #$08 zum Ausgeben des nächsten Bytes. Ansonsten hat das Maschinenprogramm ein vollständiges Zeichen an den Drucker gesandt und damit seinen Dienst beendet. Der Rücksprung in das aufrufende Programm, das in unserem Fall ein BASIC-Programm sein wird, erfolgt.

Listing Nr. 3 gibt das soeben besprochene Maschinenprogramm wieder. Darin lassen sich auch die Sprungadressen verfolgen. Damit nicht der Maschinenteil und der noch zu erstellende BASIC-Teil getrennt geladen werden müssen, empfiehlt sich die Wandlung in DATA-Zeilen mit dem DATA-WANDLER. Wir haben dies bereits in Form von Listing Nr. 4 für Sie erledigt.

HARDCOPYROUTINE

Da ein Großteil der Anwender wohl einen COMMODORE-Drucker oder weitgehendst Kompatiblen ihr eigen nennen werden, soll als erstes die Hardcopy mit einem grafikfähigen Commodore-Drucker besprochen werden. Hierfür ist das Maschinenprogramm in einigen wenigen Punkten zu modifizieren, was mit POKE-Befehlen bewerkstelligt werden kann.

POKE 321,106

Damit wird der ROL-Befehl an Speicherstelle $0141 durch den Code für den ROR-Befehl überschrieben. Der Commodore-Drucker steuert im Grafikbetrieb nur sieben Nadeln an. Mit unserer einfachen Hardcopy werden wir wohl oder übel auf die achte Linie jedes Zeichens verzichten müssen. Zu ihrem Druck bedarf es eines aufwendigeren Programmes, das bei der nächsten zu druckenden Zeile das oder die übrig gebliebenen Bytes der vorhergehenden Bildschirmzeile mit übernimmt. Dafür gibt es bereits HIRES-HARDCOPY aus dem C16/P4-SPECIAL Nr. 1. Wir dürfen beim Commodore-Drucker nicht vergessen, Bit sieben des an den Drucker auszugebenden Codes zu setzen.

POKE 325,9:POKE 326, 128

Der Operationscode für ORA #$ entspricht dem Zahlenwert 9. Mit diesem Wert ist der NOP-Befehl an Adresse $0145 zu

```
10 rem listing 9======(hardcopies) <nb>
20 rem by alfons mittelmeyer       <if>
30 rem ==========================  <ng>
40 rem fuer druckdichte von        <eo>
50 rem 240 dots/zoll.              <he>
60 rem maschinensprachteil         <pj>
70 rem ==========================  <po>
80 fori=312to366                   <ih>
90 reada:pokei,a:next              <dk>
100 data 032,078,001,160,008,162   <mm>
110 data 008,022,210,042,202,208   <gd>
120 data 250,234,234,032,102,001   <kd>
130 data 136,208,240,096,160,000   <ej>
140 data 133,210,162,008,177,208   <dg>
150 data 149,210,005,210,133,210   <im>
160 data 230,208,208,002,230,209   <jg>
170 data 202,208,239,096,032,210   <im>
180 data 255,032,210,255,076,210   <ae>
190 data 255                       <gi>
200 rem ==========================  <ne>
210 rem  basic-teil ist anzufuegen <ac>
220 rem ==========================  <hm>
```

```
10 rem centronic============plus4  <bb>
20 rem (p) commodore welt team     <ho>
30 rem ==========================  <ng>
40 rem (c) by alfons mittelmeyer   <cg>
50 rem                             <pd>
60 rem besonders geeignet fuer     <bd>
70 rem grafikprogramme, da keine   <bd>
80 rem codewandlung stattfindet    <ni>
90 rem ==========================  <jg>
100 fori=1015 to 1068              <dg>
110 reada:pokei,a:next             <ne>
120 data 72,165,153,201,3,208,43   <pp>
130 data 165,173,201,116,208,37    <fh>
140 data 104,76,19,4,72,32,247,3   <in>
150 data 104,96,10,32,19,4,104,141 <pb>
160 data 16,253,169,0,141,2,253    <gh>
170 data 169,8,141,2,253,169,32,44 <pp>
180 data 1,253,240,251,24,96,104   <ne>
190 data 76,75,236,166,174,224,4   <in>
200 data 208,8,162,3,134,174,162   <on>
210 data 116,134,173,76,166,6,166  <nk>
220 data 173,224,116,208,4,162,0   <fc>
230 data 134,173,32,12,239,162,0   <gl>
240 data 134,173,96,32,83,239,76   <oj>
250 data 161,6,32,93,238,76,161,6  <pk>
260 fori=1667 to 1713              <mc>
270 reada:pokei,a:next             <jj>
280 poke792,131:poke793,6          <gh>
290 poke800,148:poke801,6          <cn>
300 poke804,008:poke805,4          <da>
310 poke794,172:poke795,6          <lo>
320 rem ========e=n=d=e=========  <cf>
```

```
10 rem centronics==============c16 <hl>
20 rem (p) commodore welt team     <ho>
30 rem ========================== <ng>
40 rem (c) by alfons mittelmeyer  <cg>
50 rem nur in verbindung mit dem  <hj>
60 rem centronics-interface aus   <oa>
70 rem c16-p4-spezial nr.1.       <pf>
80 rem besonders geeignet fuer    <bk>
90 rem grafik-programme           <ak>
100 rem ========================= <id>
110 fori=1026to1067               <pf>
120 reada:pokei,a:next            <ag>
130 data 072,133,245,165,001,162  <dd>
140 data 008,009,003,006,245,144  <kp>
150 data 002,041,253,133,001,041  <hi>
160 data 254,133,001,202,208,239  <np>
170 data 009,004,133,001,041,249  <ml>
180 data 133,001,036,001,080,252  <np>
190 data 024,174,001,004,104,096  <el>
200 fori=1659to1729               <bh>
210 reada:pokei,a:next            <ap>
220 data 072,165,153,201,003,208  <gh>
230 data 060,165,173,201,116,208  <bd>
240 data 054,104,142,001,004,076  <al>
250 data 002,004,166,174,224,004  <oo>
260 data 208,008,162,003,134,174  <ll>
270 data 162,116,134,173,076,178  <ig>
280 data 006,166,173,224,116,208  <of>
290 data 004,162,000,134,173,032  <gl>
300 data 012,239,162,000,134,173  <lj>
310 data 096,032,083,239,076,173  <af>
320 data 006,032,093,238,076,173  <nl>
330 data 006,104,076,075,236      <go>
340 fori=818to858                 <ai>
350 reada:pokei,a:next            <di>
360 data 169,160,141,032,003,169  <ni>
370 data 006,141,033,003,169,143  <dd>
380 data 141,024,003,169,006,141  <ef>
390 data 025,003,169,184,141,026  <ld>
400 data 003,169,006,141,027,003  <ag>
410 data 169,123,141,036,003,169  <aa>
420 data 006,141,037,003,096      <jn>
430 sys818:new                    <lc>
440 rem ========================= <kf>
450 rem  p r o g r a m m e n d e  <pc>
460 rem ========================= <jk>
```

überschreiben. Das Byte im Akku ist mit dem Wert $80 zu odern, damit Bit sieben gesetzt wird. Erreicht wird dies durch POKEn des Wertes 128, der die dezimale Entsprechung zu $80 ist, in die nachfolgende Speicherzelle $0146 (dezimal 326). Das Maschinenprogramm ist jetzt an den Commodore-Drukker angepaßt.

INITIALISIERUNG DES ZEIGERS

Das Maschinenprogramm verwendet den Inhalt der Adressen $D0 und $D1 als Zeiger zum Lesen des jeweils nächsten Zeichens. Vor dem Start der Hardcopy ist der Zeiger auf den Anfang des Grafikbereiches $2000 einzustellen.

POKE 208,0:POKE 209, 32

Dezimal ergibt $D0 den Wert 208. Der in die folgende Speicherstelle zu POKEnde Wert $20 ergibt dezimal 32.

ÖFFNEN DES DRUCKERKANALS

Der Drucker muß mit dem OPEN-Befehl als Datei angemeldet sein. Damit er als Ausgabegerät auch für die BSOUT-Routine des Maschinenprogramms bereitsteht, ist der Druckerkanal mit dem CMD-Befehl zu öffnen.

OPEN 4,4:CMD4, CHR$(8);

Folgt auf den CMD-Befehl ein Komma, so wird der Rest danach genau wie bei einem PRINT-Befehl ausgegeben. Mit CHR$(8) stellen wir den Drucker auf Grafik um. Codes, die größer oder gleich 128 sind, Bytes mit gesetztem siebten Bit also, steuern jetzt direkt die obersten sieben Nadeln an.
Gleichzeitig erfolgt der Zeilenvorschub in kürzeren Abständen, so daß die nächste Druckzeile ohne Zwischenraum an die vorhergehende anschließt.

AUSGABE AN DEN DRUCKER

Die Ausgabe gestaltet sich recht einfach. Es sind 25 Zeilen zu jeweils 40 Zeichen an den Drucker zu senden.

FORJ=1TO25:PRINT
FORI=1TO40:SYS312:
NEXT:NEXTJ

Vor jeder Zeile bewirken wir mit PRINT einen Zeilenvorschub. Mit SYS 312 wird das Maschinenprogramm an Adresse $0138 (dezimal 312) aufgerufen, das genau ein Zeichen ausgibt.

PRINT#4,CHR$(15)
:CLOSE4

Mit PRINT#4,CHR$(15) schalten wir den Drucker wieder auf Text um, geben einen zusätzlichen Wagenrücklauf und Zeilenvorschub aus, da kein Strichpunkt folgt, und schließen den Druckerkanal. Die CLOSE-Anweisung beseitigt den Eintrag des logischen Files. Listing Nr. 5 enthält den BASIC-Teil der Hardcopy für grafikfähige Commodore-Drucker. Er ist an den in DATA-Zeilen gewandelten Maschinenteil anzubinden.

IBM- und Epson-Drucker können mit acht Nadeln drucken. Es braucht daher keine Linie unterschlagen zu werden. Da für den Acht-Nadel-Drucker jedes Bit des zu sendenden Bytes zur Nadelansteuerung dient, ergibt sich die Frage, wie wieder in den Normalmodus umgeschaltet werden kann. Durch einen Steuercode wohl kaum, denn jeder Code wird als Bitmuster verstanden. So wird hier ein etwas anderer Weg eingeschlagen. Den Steuercodes zum Umschalten auf Grafik folgen zwei Bytes, die angeben, wieviel Daten-Bytes darauf im Grafikmodus ausgegeben werden sollen. Nach Ausgabe dieser Anzahl kann der Drucker wieder ASCII-Codes und Steuercodes interpretieren. Mehr Flexibilität erlauben IBM und Epson, da der Zeilenabstand frei wählbar ist.

POKE 208,0:POKE 209, 32
E$=CHR$(27):X$=
CHR$(13)+CHR$(10)

Der Adreßzeiger muß wieder auf den Grafikanfang gesetzt werden. Das Programm wird kürzer und übersichtlicher,

wenn mehrfach auftretende Steuercodes als Stringvariablen definiert werden. Ist der automatische Zeilenvorschub bei Wagenrücklauf eingestellt, so darf X$ nicht zusätzlich den Steuercode für Linefeed enthalten. X$=CHR$(13) darf es in diesem Fall nur heißen.

OPEN4,4:CMD4,E$"A" CHR$(8);

Mit Escape und A wird der Zeilenabstand gewählt. Das folgende Byte n gibt diesen in n/72 Zoll wieder. Für n können Werte von Null bis 85 gewählt werden. Da der Abstand der Nadeln voneinander 1/72 Zoll beträgt und mit acht Nadeln gedruckt wird, wählten wir den dafür richtigen Zeilenabstand 8/72 Zoll.

PRINTE$"2";

Dieser Steuercode ist nur für den IBM-Drucker erforderlich. Der vorher angewählte Zeilenabstand wird erst durch diesen Befehl aktiviert, beim Epson-Drucker hingegen sofort.
ESC und "2" würden den Zeilenabstand beim Epson-Drucker auf 1/6 Zoll stellen, was wir hier nicht brauchen können.

FORJ=1TO25 PRINTXE"K"CHR$ (64)CHR$(1);

Mit X$ wird der Zeilenvorschub realisiert. Danach ist dem Drucker zu melden, daß er in Grafik mit einer bestimmten Dichte drucken soll. Vier verschiedene Dichten stehen für den IBM-Drucker zur Auwahl. Der Epson-Drucker kennt darüber hinaus noch fünf weitere. Die Druckpunkte, die der Drucker zu Papier bringt, heißen nicht Pixel wie beim Bildschirm, sondern Dots. Nachstehend finden Sie die Anwahl der vier Dichten, die sowohl IBM- als auch Epson-Drucker kennen.

ESC K = 60 Dots/Zoll
ESC L = 120 Dots/Zoll
ESC Y = 120 Dots/Zoll
ESC Z = 240 Dots/Zoll

Mit ESC Y erfolgt der Ausdruck in doppelter Geschwindigkeit als mit ESC K. Der Nachteil hierbei ist, daß direkt nebeneinander liegende Druckpunkte unterschlagen werden. Von zwei solchen Punkten, die schwarz zu Papier kommen sollten, bleibt einer also weiß.
Die Anzahl der horizontal zu druckenden Punkte pro Zeile wird im Anschluß an die Dichteauswahl mit Hilfe zweier Bytes angegeben. Low-Byte 64 und High-Byte eins sind 64+256=320, was 40 Zeichen mit je acht Pixeln entspricht.

FORI=1TO40:SYS312 :NEXTI:NEXTJ PRINTE$"A"CHR$(12);

Das Ende des Programms vollzieht sich analog zur Hardcopy mit dem Commodore-Drucker. Mit ESC A und der Größenangabe stellen wir beim IBM-Drucker 12/72 = 1/6 Zoll Zeilenabstand ein, was der Standard für Textdruck ist. Beim Epson-Drucker kann entweder dieser Befehl oder der folgende entfallen.

PRINTE$"2";

Dieser Befehl aktiviert den eingestellten Zeilenabstand beim IBM-Drucker. Beim Epson-Drucker stellt er immer den Standardabstand von 1/6 Zoll ein.

PRINT#4,X$;:CLOSE4

Wir schließen nach dem Senden von Wagenrücklauf und Zeilenvorschub den Druckerkanal. Mit CLOSE wird der Eintrag in die File-Tabelle beseitigt.
Zusammengefaßt finden Sie die Programme für IBM- und Epson-Drucker als Listing Nr. 6 und Nr. 7 im Anhang.

UNIVERSAL-HARDCOPY AUCH FÜR NICHT GRAFIKFÄHIGE DRUCKER

Eine Hardcopy für nicht grafikfähige Drucker, das gibt es doch nicht, wird manch einer sagen. Wir haben uns ein Verfahren überlegt, das sowohl auf nicht grafikfähigen Commodore-Druckern funktioniert als auch mit den vielen nur teilweise Commodore-kompatiblen Druckern zusammenarbeitet, die vielleicht den Textmodus von Commodore noch richtig emulieren, deren Grafik aber anders anzusteuern ist. Normalerweise wird es Epson- oder IBM-Grafik sein, kann aber auch andersartig angesteuert werden.

POKE 208,0:POKE209, 32 OPEN5,4,5:OPEN4,4: PRINT#4,CHR$(8);

Die letzte Zeile bringt an den Tag, was wir vorhaben. Mit Sekundäradresse fünf läßt sich ein Zeichen, das sogenannte Geschäftszeichen, vom Benutzer frei definieren. Mit dem Code CHR$ (254) kann dieses Geschäftszeichen dann unter anderer Sekundäradresse zu Papier gebracht werden. Wenn ein nicht grafikfähiger Drukker allerdings mit CHR$ (8) nicht einmal einen kürzeren Zeilenabstand einstellen kann, sieht die Hardcopy ein wenig gestückelt aus, da weiß gebliebene Linien den Ausdruck zerteilen.
Ein Problem gilt es noch zu lösen. Die Definition des Geschäftszeichens darf nur am Zeilenanfang erfolgen und gilt dann für die ganze Zeile. Im Druckerhandbuch steht, daß verschiedene Geschäftszeichen nur durch mehrfaches Überdrucken derselben Zeile möglich seien.

FORJ=1TO25:PRINT#4 FORI=1TO40:SYS334 IFPEEK(210)=0THEN...

Mit SYS334 werden die Daten nur gelesen, und es wird erfaßt, ob überhaupt etwas zu drucken wäre. Ist dies nicht der Fall, was mit dem PEEK-Befehl festzustellen ist, so überspringen wir die eigentliche Druckerroutine. Nötig wäre dieses Vorgehen nicht, doch ersparen wir dadurch dem Drucker Arbeit. Grafiken, die mehrere Leerzeilen und viele leere Zeichen enthalten, kommen so schneller auf das Papier.

PRINT#4,CHR$(141);: CMD5,;:SYS315

Mit CHR$(141) erfolgt ein Wagenrücklauf ohne Zeilenvorschub. Nachdem der Druckkopf sich am Zeilenanfang befindet, kann auf Sekundäradresse fünf umgeschaltet und mit SYS315 das Zeichenmuster gesendet werden. Das Komma und der Strichpunkt bei der CMD-Anweisung sind wichtig, damit der Drukker nicht als erstes Bitmuster den Return-Code erhält. Das letzte Bitmuster würde den Drucker außerdem in Verwirrung bringen.

PRINT#5:A$=RIGHT$ ("0"+STR$(I–1),2) PRINT#4,CHR$(16)A$ CHR$(254); NEXT:NEXTJ

Mit PRINT#5 wird die Zeichendefinition beendet. Die Tabulatorfunktion CHR$(16) verwenden wir, um den Druckkopf auf die richtige Stelle in der Zeile zu fahren.

Die Position muß im Anschluß durch eine zweistellige ASCII-Zahl angegeben werden. Die Berechnung erfolgt durch die STR$-Funktion. Ist das Ergebnis nur einstellig, so setzen wir die Ziffer Null davor. Die RIGHT$-Funktion sorgt in jedem Fall für die richtige Län-

ge. Nach der Positionsangabe A$ brauchen wir nur noch den Code CHR$ (254) zu senden, um das definierte Zeichen zu Papier zu bringen.
Bei der Definition des Zeichens fährt der Druckkopf nicht bis zum Anschlag zurück, da der Drucker so intelligent ist, zu warten, ob nicht etwa wieder eine gegenläufige Bewegung erfolgen soll. Dennoch ruckelt der Druckkopf bei den meisten Druckern ganz schön hin und her. Aus diesem Grunde hatten wir auch für eine teilweise Entlastung gesorgt.

PRINT#4,CHR$(15): CLOSE4:CLOSE5

Das Beenden der Routine erfolgt wie gewohnt. Das Programm finden Sie als Listing Nr. 8 im Anhang.

EXOTENDRUCKER MCS801

Obwohl der Farbdrucker MCS801 auch ein Commodore-Drucker ist, ist er, was Grafik angeht, keineswegs kompatibel.

GESCHÄFTSZEICHEN-MODUS

Das Universalprogramm für nicht grafikfähige Drucker sollte doch eigentlich auch für den MCS801 Gültigkeit haben, da hier der Geschäftszeichenmodus genutzt wird. Doch ist dies eine Täuschung. Die Zeichen werden umgedreht wiedergegeben. Was zu tun ist, wissen wir aber bereits. Mit POKE 321, 106 sorgen wir für die richtige Wiedergabe.

GRAFIKMODUS

Beim Drucken im Grafikmodus ist zu beachten, daß dieser ähnlich wie bei Epson oder IBM angesteuert wird. Jedoch ist die Befehlssequenz eine andere. Die Anzahl der zu druckenden Punktreihen pro Zeile ist nicht im Byte-Format anzugeben, sondern als dreistellige ASCII-Zahl. Damit würde die Sequenz am Anfang einer Druckzeile lauten:

PRINTXE"K320";

Epson- und IBM-Drucker erfordern dagegen:

PRINTXE"K"CHR$ (64)CHR$(1);

Einfacher und einprägsamer ist für den Anwender sicherlich die Ansteuerung des MCS-801. Auf einen Blick ist zu sehen, daß horizontal 320 Punkte gedruckt werden sollen. Zum Nachschlagen, wie der richtige Zeilenabstand von 8/72 Zoll eingestellt wird, verwenden Sie bitte Ihr Druckerhandbuch.

WARUM DER KREIS ZUR ELLIPSE WIRD

Auf das richtige Verhältnis von Höhe und Breite kommt es an. Wenn es nicht stimmt, ist nicht der Computer daran schuld, sondern der Drucker. Genauer gesagt, die Software, die den Drucker ansteuert. Meist läßt sich für die richtige Wiedergabe etwas tun. Wählen wir die normale Grafikdichte von 60 Dots/Zoll, so wird der Ausdruck zu breit. Der vertikale Nadelabstand beträgt nämlich 1/72 Zoll, was einer Dichte von 72 Dots/Zoll entspricht. Beim Commodore-Drucker im Grafikmodus gilt somit das Verhältnis:

Breite/Höhe=1.2

In der in diesem Artikel vorgestellten Hardcopy, bei der wir eine Linie pro Zeile wegließen, ist es gar noch schlechter:

8/60*72/7=1.37

Auch für den Commodore-Drucker gibt es die Möglichkeit, Breite und Höhe im Verhältnis eins zu eins darzustellen. Wenn Sie die Hardcopy für den Geschäftszeichenmodus ausprobiert haben, so sollten Sie es bereits beobachtet haben. Im normalen Schriftmodus gibt der Drucker zehn Zeichen pro Zoll aus. Ist nicht NLQ gewählt, wird das Zeichen im Normalfall mit einer Dichte von 60 Dots/Zoll gedruckt, so daß in der Breite nur sechs Punkte dafür zur Verfügung stehen. Bei einigen Druckern wie dem MCS-801 kann im Schriftmodus auch eine größere Dichte vorliegen, weswegen vielleicht bei ihnen die Grafik-Hardcopy bereits richtig kommen könnte.
Dieselbe Dichte bekommen wir beim normalen Drucker mit dem definierbaren Geschäftszeichen. Dieses Zeichen ist ebenfalls ein zehntel Zoll breit, besteht jedoch aus acht horizontalen Punkten. Die Druckdichte beträgt somit 80 Dots pro Zoll. Bei 7/72 Zoll Zeilenabstand wird die unterste Linie einer Zeile nochmals mit der obersten der nächsten Zeile überdruckt. Unser Zeichen ist also 8/80 Zoll breit und 7/72 Zoll hoch.

8/80*72/7=1.03

Der Dehnungsfaktor von 1.03 fällt überhaupt nicht ins Auge. Damit ist für Commodore-Drucker alles klar.

EPSON-DRUCKER BESTENS AUSGESTATTET

Keinerlei Klimmzüge braucht es beim Epson-Drucker. Speziell für Hardcopies und Bildschirmplots hat dieser auch eine Druckdichte von 72 Dots/Zoll, wodurch vertikaler und horizontaler Dot-Abstand genau gleich sind. Die Befehlssequenz, um weitere Druckdichten als es der IBM-Drucker erlaubt, einzustellen, lautet:

ESC * m nl n2

Der Wert von m legt dabei die Druckdichte fest.

m	Dichte
0	60 Dots/Zoll
1	120 Dots/Zoll
2	120 Dots/Zoll
3	240 Dots/Zoll
4	80 Dots/Zoll
5	72 Dots/Zoll
6	90 Dots/Zoll

Die ersten vier Dichten von m=0 bis m=3 entsprechen den bereits bekannten, die auch durch ESC K, ESC L, ESC Y und ESC Z erzeugt werden können. Die restlichen Dichten sind für Grafik-Hardcopies mit verschiedenen Rechnern vorgesehen. Einige Beispiele:

Commodore 72 Dots/Zoll
Epson-QX-10 80 Dots/Zoll
DEC-Computer 90 Dots/Zoll

Auf m als Dichteparameter muß auch hier wieder die Anzahl der zu druckenden Datenbytes folgen. Wir ändern im Epson-Listing daher:

PRINTXE"*"chr$(5) chr$(64)chr$(1);

TRICK FÜR IBM-DRUCKER

Über die Druckdichte 72 Dots/Zoll verfügt der IBM-Drucker nicht. Wenn wir 80 Dots/Zoll erreichen könnten, könnten wir dasselbe Verfahren anwenden wie bereits bei den Commodore-Druckern. Ein Trick hilft hier weiter: Wir stellen die Druckdichte auf 240 Dots/Zoll und geben jedes Datenbyte dreimal hintereinander aus. Den Zeilenabstand stellen wir außerdem auf 7/72 Zoll.

OPEN4,4:CMD4,ES"A" CHRS(7); PRINTXSES"Z"CHRS (192)CHRS(3);

Welche Zeilen im IBM-Listing zu ersetzen sind,

dürfte unschwer zu erkennen sein. Die dreimalige Ausgabe jedes Datenbyte wird am besten im Maschinenprogramm vorgenommen. Der Ausgabebefehl BSOUT, JSR $FFD2 wird einfach durch einen Sprung in eine Unterroutine ersetzt, in welcher steht:

```
JSR $FFD2
JSR $FFD2
JMP $FFD2
```

Diese neue Maschinenroutine finden Sie als Listing Nr. 9 im Anhang.

KLEINER, GRÖSSER, DICHTER

Die Punkte, die bei der vorangegangenen Hardcopy dicht beieinanderliegen, lassen die Zwischenräume fast verschwinden. Wenn der vertikale Nadelabstand ebenfalls verringert würde, so würden schwarze Flächen ihren Pünktchencharakter verlieren und beinahe als gefüllte Flächen wirken. Eine solche Hardcopy sieht wesentlich besser aus.
Durch mehrfaches Drucken von Datenbytes und Zeilen lassen sich vergrößerte Hardcopies erzielen. Größere Dichte und Zeilenabstände von 1/216 Zoll machen Verkleinerungen möglich. Mehr darüber gibt es im nächsten SPECIAL, im zweiten und letzten Teil dieser Folge.
Probleme können bei Centronics-Druckern mit Commodore-Interface auftreten. Ein gutes Interface läßt die Umschaltung in den Epson- oder IBM-Modus zu. Ein schlechtes arbeitet bei Grafik vielleicht fehlerhaft oder erlaubt nur den Commodore-Modus. Wer alle Möglichkeiten seines Druckers ausschöpfen möchte, sei nochmals auf das Centronics-Kabel für den Plus/4 und an das Centronics-Interface für den C16 hingewiesen, worüber wir in der COMMODORE WELT Nr. 12/88 und im C16-P4-SPECIAL Nr. 1 berichteten. Das Centronics-Kabel gibt es mittlerweile bei Rex-Datentechnik, die Bezugsadresse für das Interface können Sie bei uns erfragen. Da die Treibersoftware für den Textmodus gedacht und mit ASCII-Wandlung, automatischem Zeilenvorschub bei Return oder Unterdrückung des ASCII-Codes 17 ausgestattet war, finden Sie im Anhang die Centronics-Software für den Grafikmodus.

Auf einen Fehler in der Beschreibung des Interface für den C16 müssen wir die Selbstbauer hinweisen. Wir verwechselten die Leitungen Strobe und Busy. Tauschen Sie daher am Druckerstecker die Leitung an Pin eins gegen die an Pin 11 aus. Pin drei des seriellen Ports muß an Pin eins des Druckerports und Pin elf des Druckerports an den Eingang des Inverters gelegt werden. Am günstigsten ist, zuerst das Programm mit der Grafik aufzurufen und es an der gewünschten Stelle zu unterbrechen. Dann ist die Centronics-Software zu laden und zu starten, danach erst die Hardcopyroutine. Vorher drucken ist ja nicht möglich. Weiter geht es im nächsten Heft mit kleineren, größeren und dichteren Hardcopies. *a.m.* □

Russisches Roulett

Bugs Bunny, der Hase, lebt von Karotten. Eine bestimmte Anzahl davon ist sein täglicher Bedarf. Der Bauer jedoch macht Bugs Bunny das Leben schwer. Er hat eine Bombe unter den Karotten plaziert.

Das Spiel Bugs Bunny besteht aus zwei Bildern. Im ersten versucht der Hase, dem Bauern die Karotten zu stehlen. Dieser legt zwölf Karotten in seinem Garten aus. Sie steuern Bugs Bunny mit dem Joystick Port 1. Haben Sie sich für eine Karotte entschieden, lenken Sie ihn so nahe wie möglich zur Karotte und drücken Sie den Feuerknopf.
Vier Karotten muß Bugs Bunny einsammeln, um Runde zwei zu erreichen. Dort hat er fünf und schließlich, in Runde drei, sechs Karotten zu klauen.
Aber Vorsicht, der Bauer legt in jeder Runde eine Karotte als Bombe präpariert aus. Nimmt Bugs Bunny die Bombe, so verliert er eines von seinen drei Leben. Jeder Rundenwechsel kündigt sich durch Musik und andere Rahmenfarbe an. Hat Bugs Bunny Runde drei geschafft, kommt das zweite Bild.
Hier versucht der Bauer, den Hasen zu fangen. Zum Versteck von Bugs Bunny führen fünf Wege. Jeder ist mit Karotten ausgelegt. Aber in zwei Wegen hat Bugs Bunny eine Bombe versteckt.
Sie steuern den Bauern mit dem Joystick und müssen sich für einen Weg entscheiden: Steuerung hoch, runter und nach rechts. Tritt der Bauer auf eine Bombe, so verliert er sein Leben und das Spiel ist verloren. Kann der Bauer Bugs Bunny erreichen, ist es gewonnen. □

```
10 rem bugs bunny=============c16  <lm>
20 rem (p) commodore welt team     <ho>
30 rem ========================    <mm>
40 rem (c) j. kienberger           <ai>
50 rem                             <pd>
60 rem                             <ah>
70 rem basic v3.5                  <nl>
80 rem c16/116/plus4               <ki>
90 rem ========================    <km>
100 clr:vol6:gosub2570             <ia>
110 poke55,0:poke56,56:clr         <lg>
120 gosub2660:scnclr:color0,5,1    <af>
130 char,12,10,wh$+"bitte warten"  <dm>
140 poke65298,peek(65298)and251    <cp>
150 poke65299,peek(65299)and3or56  <hc>
160 forb=0to17:reada               <cg>
170 poke1630+b,a:next:sys1630      <ep>
180 fork=14856to15008step8         <lg>
190 forb=0to7:reada                <en>
200 ifa=-1then430                  <ee>
210 pokek+b,a:nextb,k              <mj>
220 data162,0,189,0,208,157,0,56,1
89,0,209,157,0,57,202,208,241,96   <op>
230 data0,0,0,0,0,195,102,60       <io>
```

```
240 data126,126,60,60,60,24,24,24   <nc>
250 data128,96,112,56,30,15,15,7   <nk>
260 data1,6,14,28,120,240,240,224   <do>
270 data7,15,15,30,56,112,96,128   <mj>
280 data224,240,240,120,28,14,6,1   <np>
290 data129,195,231,102,60,24,24,2
4   <be>
300 data8,12,6,6,3,7,5,7   <gm>
310 data16,48,96,96,192,224,160,22
4   <mn>
320 data7,6,3,1,3,15,31,55   <dn>
330 data224,96,192,128,192,240,248
,252   <lc>
340 data55,7,7,6,6,6,6,14   <ga>
350 data236,224,224,96,96,96,96,11
2   <bk>
360 data7,7,31,8,10,5,4,3   <ej>
370 data224,224,248,16,80,160,32,1
92   <jb>
380 data7,31,63,103,199,7,7,7   <ii>
390 data224,248,252,230,227,224,22
4,224   <bb>
400 data7,7,7,6,6,6,6,14   <aj>
410 data224,224,224,96,96,96,96,11
2   <ij>
420 data-1   <mg>
430 a$=gr$+"A"+c4$+c1$+oe$+"B":b$=
ye$+"CD"+c4$+c1$+c1$+"EF"   <eo>
440 c$=gr$+"G":e$=wh$+"HI"+c4$+c1$
+c1$+"JK"+c4$+c1$+c1$+"LM"   <cm>
450 d$=ye$+"NO"+c4$+c1$+c1$+bl$+"P
Q"+c4$+c1$+c1$+re$+"RS"   <cj>
460 a$(1)=b3$+c4$+c1$+b3$:b$(1)=b2
$+c4$+c1$+c1$+b2$   <al>
470 e$(1)=b2$+c4$+c1$+c1$+b2$+c4$+
c1$+c1$+b2$   <cj>
480 rem -----------------------   <jc>
490 rem  bildschirmaufbau   <gl>
500 rem -----------------------   <dg>
510 scnclr:vol8:color0,1,1:poke652
86,11   <lb>
520 fort=23to1step-1:char,38,t,g3$
+"H":char,36,t,g3$+"H":char,37,t,g
3$+"H"   <df>
530 char,34,t,g3$+"H":char,32,t,g3
$+"H":char,11,t,g3$+"H":char,33,t,
g3$+"H"   <io>
540 char,10,t,"H":char,35,t,g3$+"H
":char,9,t,g3$+"H":next   <cn>
550 char,1,2,wh$+"karotten":char,2
,10,wh$+"leben":char,2,18,wh$+"run
de"   <gb>
560 fort=1to7:char,t,3,re$+"B":cha
r,t,7,re$+"B":char,t,11,ye$+"A"   <bl>
570 char,t,15,ye$+"A":char,t,19,lg
$+"G":char,t,23,lg$+"G":next   <po>
580 char,1,5,re$+"B":char,7,5,re$+
"B"   <ng>
590 char,1,13,ye$+"A":char,7,13,ye
$+"A"   <dd>
600 char,1,21,lg$+"G":char,7,21,lg
$+"G":poke65286,27   <kk>
610 l=3:lv=1:v1=0   <ho>
620 k=4+v1:char,3,21,wh$+str$(lv):
char,3,13,wh$+str$(l):char,3,5,wh$
+str$(k)   <pk>
630 char,16,21,b2$+b2$+b2$+b2$+b2$
+b2$+b2$+b2$   <ij>
640 z=8:x=20:s1%=2:t1%=2:v1%=2:u1%
=2:w1%=2:x1%=2   <dk>
650 s2%=2:t2%=2:v2%=2:u2%=2:w2%=2:
x2%=2   <ff>
660 fort=1to700:next   <op>
670 r1=29:r2=14   <oe>
680 rem -----------------------   <gb>
690 rem   karotten setzen   <jp>
700 rem -----------------------   <jm>
710 char,r1,22,a$:char,27,21,d$:fo
rt=1to500:next:char,27,21,e$(1)   <lb>
720 x=x-1:ifx<1then810   <di>
730 ifx=17thengosub800   <oi>
740 ifx=13thengosub800   <gc>
750 ifx=9thengosub800   <ol>
760 ifx=5thengosub800   <ah>
770 ifx=1thengosub800   <oa>
780 char,27,x,d$:char,27,x+3,e$(1)
:sound2,600,0   <el>
790 goto720   <ep>
800 char,r1,x+1,a$:sound1,800,2:re
turn   <ld>
810 char,27,x+1,e$(1):fort=1to400:
next   <eg>
820 w=4:char,15,1,d$:char,r2,2,a$:
fort=1to500:next:char,15,1,e$(1)   <nc>
830 w=w+1:ifw>21then920   <co>
840 ifw=5thengosub910   <fm>
850 ifw=9thengosub910   <mg>
860 ifw=13thengosub910   <ml>
870 ifw=17thengosub910   <ol>
880 ifw=21thengosub910   <pg>
890 char,15,w-3,e$(1):char,15,w,d$
:sound2,600,0   <pe>
900 goto830   <dk>
910 char,r2,w+1,a$:sound1,800,2:re
turn   <ci>
920 char,15,w-1,e$(1)   <hi>
930 fort=1to400:next   <ad>
940 a=int(rnd(0)*12)+1   <kh>
950 ifa=1thenc=15:o=1   <on>
960 ifa=2thenq=27:o=1   <on>
970 ifa=3thenc=15:o=5   <hd>
980 ifa=4thenq=27:o=5   <oj>
990 ifa=5thenc=15:o=9   <pg>
1000 ifa=6thenq=27:o=9   <ga>
1010 ifa=7thenc=15:o=13   <bd>
1020 ifa=8thenq=27:o=13   <cm>
```

```
1030 ifa=9thenc=15:o=17                    <lo>
1040 ifa=10thenq=27:o=17                   <eh>
1050 ifa=11thenc=15:o=21                   <mj>
1060 ifa=12thenq=27:o=21                   <oo>
1070 u1=21:u2=21                           <gj>
1080 rem -----------------------           <he>
1090 rem      steuerung                    <hn>
1100 rem -----------------------           <fj>
1110 poke1344,64:poke239,0                 <nb>
1120 char,u1,u2,e$                         <lh>
1130 ifjoy(1)=1thengosub1190               <fk>
1140 ifjoy(1)=5thengosub1220               <pa>
1150 ifjoy(1)=3thengosub1250               <np>
1160 ifjoy(1)=7thengosub1280               <cd>
1170 ifjoy(1)>127thengosub1310             <ma>
1180 goto1120                              <ip>
1190 u2=u2-4:ifu2<1thenu2=1                <hb>
1200 char,u1,u2+4,e$(1)                    <ej>
1210 return                                <al>
1220 u2=u2+4:ifu2>21thenu2=21              <mo>
1230 char,u1,u2-4,e$(1)                    <ne>
1240 return                                <eh>
1250 u1=u1+1:ifu1>27thenu1=27              <pi>
1260 char,u1-1,u2,e$(1)                    <bo>
1270 return                                <id>
1280 u1=u1-1:ifu1<15thenu1=15              <km>
1290 char,u1+1,u2,e$(1)                    <jh>
1300 return                                <ma>
1310 ifu1=qandu2=othen1530                 <lg>
1320 ifu1=candu2=othen1590                 <el>
1330 ifu1=27then1360                       <gl>
1340 ifu1=15then1430                       <lc>
1350 return                                <ce>
1360 ifu2=1thens1%=s1%-1:ifs1%<1th
enreturn                                   <dj>
1370 ifu2=5thent1%=t1%-1:ift1%<1th
enreturn                                   <ec>
1380 ifu2=9thenu1%=u1%-1:ifu1%<1th
enreturn                                   <cj>
1390 ifu2=13thenv1%=v1%-1:ifv1%<1t
henreturn                                  <fn>
1400 ifu2=17thenw1%=w1%-1:ifw1%<1t
henreturn                                  <gm>
1410 ifu2=21thenx1%=x1%-1:ifx1%<1t
henreturn                                  <lk>
1420 char,r1,u2+1,a$(1):sound2,800
,4:sound2,950,3:k=k-1:goto1500             <ol>
1430 ifu2=1thens2%=s2%-1:ifs2%<1th
enreturn                                   <ei>
1440 ifu2=5thent2%=t2%-1:ift2%<1th
enreturn                                   <fp>
1450 ifu2=9thenv2%=v2%-1:ifv2%<1th
enreturn                                   <fp>
1460 ifu2=13thenu2%=u2%-1:ifu2%<1t
henreturn                                  <ik>
1470 ifu2=17thenw2%=w2%-1:ifw2%<1t
henreturn                                  <kc>
1480 ifu2=21thenx2%=x2%-1:ifx2%<1t
henreturn                                  <fm>
1490 char,r2,u2+1,a$(1):sound2,800
,4:sound2,950,3:k=k-1                      <eb>
1500 char,3,5,wh$+str$(k)                  <dk>
1510 ifk<1thenlv=lv+1:gosub1650:go
sub1770:v1=v1+1:goto620                    <ae>
1520 goto1120                              <cb>
1530 char,u1+1,u2+1,b$:fort=250to1
0step-20:sound3,740+t,3:next:fort=
1to300:next                                <en>
1540 char,u1+1,u2+1,b$(1):char,u1,
u2,e$(1):l=l-1                             <jk>
1550 char,3,13,wh$+str$(l)                 <ga>
1560 ifl=0then1830                         <ll>
1570 fort=1to400:next                      <an>
1580 gosub1770: goto620                    <le>
1590 char,u1-1,u2+1,b$:fort=250to1
0step-20:sound3,740+t,3:next:fort=
1to300:next                                <hc>
1600 char,u1-1,u2+1,b$(1):char,u1,
u2,e$(1):l=l-1                             <lp>
1610 char,3,13,wh$+str$(l)                 <lo>
1620 ifl=0then1830                         <dp>
1630 fort=1to400:next                      <lk>
1640 gosub1770:goto620                     <mh>
1650 restore2640                           <mc>
1660 reads,e                               <en>
1670 ife=0then1700                         <fl>
1680 sound1,s,e                            <dp>
1690 goto1660                              <kl>
1700 bb=int(rnd(0)*14)+2:color4,bb
,2                                         <jl>
1710 return                                <ph>
1720 fort=8to0step-1:volt:fori=1to
3:sound2,750,2:nexti:sound2,900,3:
nextt                                      <dp>
1730 gosub1770                             <oc>
1740 char,4,21,wh$+str$(lv)                <ng>
1750 iflv>3then1920                        <ap>
1760 fort=1to600:next:goto620              <jj>
1770 fort=2to22step2:char,14,t,a$(
1):char,29,t,a$(1):next                    <fj>
1780 char,u1,u2,e$(1)                      <pe>
1790 fort=1to400:next                      <hm>
1800 char,2,21,wh$+str$(lv)                <bj>
1810 iflv>3then1920                        <nk>
1820 return                                <ne>
1830 fort=200to0step-10:sound2,800
+t,7:next                                  <ln>
1840 char,16,21,wh$+"noch einmal(j
/n)"                                       <hg>
1850 poke239,0:getkeys$                    <ho>
1860 ifs$="j"thenrun                       <no>
1870 ifs$="n"then1880                      <mp>
1880 scnclr:end                            <ip>
1890 rem -----------------------           <ef>
1900 rem       bild 2                      <pi>
1910 rem -----------------------           <co>
```

```
1920 scnclr:poke65286,11:color0,1,
1                                   <ob>
1930 char,2,10,wh$+"leben":char,2,
18,wh$+"bild"                       <jm>
1940 fort=1to7:char,t,11,ye$+"B":c
har,t,15,ye$+"B":char,t,19,re$+"H"  <gl>
1950 char,t,23,re$+"H":next         <co>
1960 char,1,13,ye$+"B":char,7,13,y
e$+"B":char,1,21,re$+"H":char,7,21
,re$+"H"                            <gl>
1970 fort=11to36:char,t,1,g3$+"T":
char,t,23,g3$+"T":next              <mb>
1980 fort=2to22:char,11,t,g3$+"T":
char,36,t,g3$+"T":next              <bf>
1990 fort=18to30step2:char,t,4,a$:
char,t,12,a$:char,t,20,a$           <hi>
2000 char,t,8,a$:char,t,16,a$:next  <ah>
2010 fort=17to31:char,t,2,c$:char,
t,6,c$:char,t,10,c$                 <ge>
2020 char,t,14,c$:char,t,18,c$:cha
r,t,22,c$:next                      <gj>
2030 char,33,11,e$                  <jl>
2040 poke65286,27                   <kg>
2050 l=1:r=2                        <pp>
2060 w%=int(rnd(0)*5)+1             <di>
2070 m%=int(rnd(0)*5)+1             <ee>
2080 ifw%=1thenv%=28:x%=3           <dj>
2090 ifw%=2thenv%=28:x%=7           <dl>
2100 ifw%=3thenv%=28:x%=11          <bj>
2110 ifw%=4thenv%=28:x%=15          <ib>
2120 ifw%=5thenv%=28:x%=19          <lh>
2130 ifw%=m%then2060                <ok>
2140 ifm%=1thenn%=25:o%=3           <fi>
2150 ifm%=2thenn%=25:o%=7           <lj>
2160 ifm%=3thenn%=25:o%=11          <nf>
2170 ifm%=4thenn%=25:o%=15          <hg>
2180 ifm%=5thenn%=25:o%=19          <ko>
2190 fort=8to0step-1:volt:fori=1to
3:sound2,750,2:nexti:sound2,900,3:
nextt                               <ff>
2200 vol8:char,3,13,wh$+str$(l):ch
ar,3,21,wh$+str$(r)                 <mg>
2210 z%=14:y%=19                    <mj>
2220 rem -----------------------    <pc>
2230 rem       steuerung            <cm>
2240 rem -----------------------    <dl>
2250 char,z%,y%,d$                  <ep>
2260 ifjoy(1)=1thengosub2300:gosub
2340                                <io>
2270 ifjoy(1)=5thengosub2300:gosub
2370                                <gg>
2280 ifjoy(1)=3thengosub2300:gosub
2400                                <hn>
2290 goto2250                       <ne>
2300 ifz%=v%andy%=x%then2430        <cf>
2310 ifz%=n%andy%=o%then2470        <bc>
2320 ifz%=33andy%=11then2510        <ie>
2330 return                         <ne>
2340 y%=y%-4:ify%<3theny%=3         <de>
2350 char,z%,y%+4,e$(1):sound1,600
,1                                  <pj>
2360 return                         <ba>
2370 y%=y%+4:ify%>19theny%=19       <ij>
2380 char,z%,y%-4,e$(1):sound1,600
,1                                  <dd>
2390 return                         <em>
2400 z%=z%+1:ifz%>33thenz%=33       <fl>
2410 char,z%-2,y%,e$(1):sound1,600
,1                                  <me>
2420 return                         <ii>
2430 char,z%,y%+1,b$:fort=250to10s
tep-20:sound3,740+t,3:next:fort=1t
o300:next                           <if>
2440 char,z%,y%+1,b$(1):l=l-1       <bg>
2450 ifl=0then2550                  <mc>
2460 goto1920                       <eo>
2470 char,z%,y%+1,b$:fort=250to10s
tep-20:sound3,740+t,3:next:fort=1t
o300:next                           <od>
2480 char,z%,y%+1,b$(1):l=l-1       <cl>
2490 ifl=0then2550                  <bf>
2500 goto1920                       <eb>
2510 fort=1to4:sound1,700,3:sound2
,800,5:next                         <di>
2520 scnclr:char,11,10,wh$+"g e w
o n n e n"                          <kc>
2530 fort=1to16:fors=0to7:color4,t
,s:nexts:nextt                      <ge>
2540 fort=1to6:sound2,900,3:sound1
,881,7:next:poke239,0:end           <aj>
2550 fort=200to0step-10:sound1,800
+t,5:next                           <ep>
2560 scnclr:char,11,10,wh$+"v e r
l o r e n":poke239,0:end            <jk>
2570 scnclr:char,11,11,ye$+"b u g
s  b u n n y "                      <ng>
2580 restore2640                    <ip>
2590 readu,a                        <kj>
2600 ifu=-1then2630                 <ij>
2610 sound1,u,a                     <kj>
2620 goto2590                       <fi>
2630 fort=1to600:next:return        <dd>
2640 data704,10,739,10,770,10,798,
10,810,15,810,10,834,10,739,10,854
,10,810,10                          <lk>
2650 data798,15,-1,0                <cd>
2660 rem nachspann ==============   <gf>
2670 rem * farbcodes/steuercodes *  <pa>
2680 wh$=chr$(005):c4$=chr$(017)    <hg>
2690 re$=chr$(028):gr$=chr$(030)    <ea>
2700 bl$=chr$(031):oe$=chr$(129)    <mb>
2710 bk$=chr$(144):lg$=chr$(153)    <ag>
2720 g3$=chr$(155):c1$=chr$(157)    <dc>
2730 ye$=chr$(158)                  <jm>
2740 b2$=" "+" ":b3$=" "            <fc>
2750 return                         <bp>
```

UTILITIES

Gewußt wie: Sieben Hilfsprogramme

Wollen Sie einem Datenverlust infolge eines Device Errors vorbeugen, Listingzeilen unsichtbar machen, den Bildschirm schonen und anderes? Hierfür steht Ihnen diese Routinensammlung zur Verfügung.

BILDSCHUTZ

Dieses Verfahren habe ich mir beim Programmieren auf dem Atari abgeguckt.

```
10 rem bildschutz============c16     <bg>
20 rem (p) commodore welt       ==    <hf>
30 rem ========================       <mm>
40 rem (c) by eckhard schulz    ==    <dc>
50 rem                          ==    <if>
60 rem                          ==    <nd>
70 rem basic v3.5               ==    <cg>
80 rem c16/116/plus4            ==    <fk>
90 rem ========================       <km>
100 data 78,a9,6e,8d,14,03,a9,06      <kh>
110 data 8d,15,03,58,20,db,ff,60      <ci>
120 data a5,fa,c9,01,f0,1f,a5,c6      <kd>
130 data c9,40,f0,04,a9,00,85,a4      <ac>
140 data a5,a4,c9,28,30,23,ad,06      <bi>
150 data ff,69,08,8d,06,ff,a9,01      <ah>
160 data 85,fa,4c,a7,06,a5,c6,c9      <od>
170 data 40,f0,0e,ad,06,ff,e9,08      <lb>
180 data 8d,06,ff,a9,00,85,fa,85      <cn>
190 data a4,4c,0e,ce,00,00,00,00      <ci>
200 :                                 <ml>
210 for a = 1630 to 1709              <pe>
220 read a$                           <kh>
230 poke a , dec (a$)                 <im>
240 b = b + dec (a$)                  <mo>
250 next a                            <cp>
260 if b < > 9327 then print "erro
r"                                    <de>
270 if b < > 9327 then go to 290      <cn>
280 print "aufruf : sys 1630"         <oe>
290 end                               <mg>
```

Wird auf dem Computer innerhalb von 2,45 Minuten keine Taste betätigt, wird der Bildschirm automatisch abgeschaltet. Nach dem Drücken einer beliebigen Taste wird er wieder eingeschaltet. Dabei geht der Bildschirminhalt nicht verloren. Das Programm ist im freien Speicherbereich an 1630 abgelegt und wird mit SYS 1630 gestartet.

SECURE

Schützen Sie Ihre BASIC-Programme mit diesem Programm. Durch Eingabe von fünf Doppelpunkten hinter der Zeilennummer wird diese Zeile geschützt. Beim Listen wird nur noch die Zeilennummer ausgegeben. Auch diese Routine belegt keinen BASIC-Speicherplatz. Aufgerufen wird sie mit SYS 1630.

```
10 rem secure================c16      <ij>
20 rem (p) commodore welt       ==    <hf>
30 rem ========================       <mm>
40 rem (c) by eckhard schulz    ==    <dc>
50 rem                          ==    <if>
60 rem                          ==    <nd>
70 rem basic v3.5               ==    <cg>
80 rem c16/116/plus4            ==    <fk>
90 rem ========================       <km>
100 gosub 550                         <bj>
110 summe = 0                         <fo>
120 for adresse = 1630 to 1861        <pd>
130 read byte$                        <hg>
140 summe = summe + dec (byte$)       <la>
150 poke adresse , dec (byte$)        <dh>
160 next adresse                      <dk>
170 if summe <> 24151 then print c
l$c4$"              data-error" :
 end                                  <fm>
180 print cl$c4$rn$" secure  secur
e  secure  secure  secure "rf$        <ph>
190 print c4$"     um zeilen zu sch
uetzen hinter der                     <eo>
200 print c4$"   zeilennummer 5 do
ppelpunkte setzen.                    <bk>
210 print c4$"      rufen sie dann
 die routine mit                      <cm>
220 print c4$"              sys 16
30 auf."                              <fb>
230 print c4$rn$" secure  secure
secure  secure  secure "rf$           <kp>
240 :                                 <bl>
250 data 20,6b,c5,ad,01,10,d0,05      <ge>
260 data ad,02,10,f0,26,a9,01,85      <nh>
270 data 57,a9,10,85,58,a0,00,b1      <ge>
280 data 57,85,5a,20,16,07,b1,57      <hb>
290 data 85,5b,20,16,07,b1,57,85      <pe>
300 data 5d,20,16,07,b1,57,85,5c      <bk>
310 data 4c,9f,06,a2,00,bd,22,07      <im>
320 data 20,d2,ff,e8,e0,14,d0,f5      <hf>
330 data 60,a5,5a,d0,04,a5,5b,f0      <lb>
340 data ea,a2,00,84,59,bd,36,07      <kl>
350 data 20,d2,ff,e8,e0,05,d0,f5      <fd>
360 data a9,20,20,d2,ff,a6,5d,a5      <ke>
370 data 5c,20,5f,a4,a4,59,a2,00      <ii>
380 data 20,16,07,b1,57,c9,3a,d0      <jn>
390 data 08,e8,e0,05,d0,f2,4c,e9      <ec>
400 data 06,a5,5a,85,57,a5,5b,85      <jd>
410 data 58,a0,00,a9,0d,20,d2,ff      <oi>
420 data 4c,75,06,20,1c,07,20,1c      <mi>
```

```
430 data 07,20,1c,07,20,1c,07,84    <ch>
440 data 59,a9,20,20,d2,ff,a4,59    <eo>
450 data a9,00,91,57,84,59,a2,00    <ml>
460 data bd,3b,07,20,d2,ff,e8,e0    <ob>
470 data 0a,d0,f5,a4,59,4c,d7,06    <oa>
480 data c8,d0,02,e6,58,60,88,d0    <lj>
490 data 02,c6,58,60,50,52,4f,54    <lf>
500 data 45,43,54,49,4f,4e,20,46    <oa>
510 data 49,4e,49,53,48,45,44,2e    <hf>
520 data 4c,49,4e,45,3a,50,52,4f    <cc>
530 data 54,45,43,54,45,44,2e,00    <hp>
540 end                             <lm>
550 rem nachspann =============     <gm>
560 rem *farbcodes/steuercodes *    <og>
570 c4$=chr$(017):rn$=chr$(018)     <lg>
580 rf$=chr$(146):cl$=chr$(147)     <ob>
590 return                          <co>
600 rem ======================      <dn>
610 rem p r o g r a m m e n d e     <ck>
620 rem ======================      <ik>
```

DEVICE ERROR

Ist es Ihnen noch nicht passiert? Da haben Sie 1.000 Daten ins selbstgeschriebene Dateiprogramm eingetragen und beim Speichern stellen Sie fest, daß Sie die Floppy nicht eingeschaltet haben. Die Folge: Der

```
10 rem device-error===========c16   <aa>
20 rem (p) commodore welt      ==   <hf>
30 rem ========================     <mm>
40 rem (c) by eckhard schulz   ==   <dc>
50 rem                         ==   <if>
60 rem                         ==   <nd>
70 rem basic v3.5              ==   <cg>
80 rem c16/116/plus4           ==   <fk>
90 rem ========================     <km>
95 gosub 60000                      <jp>
100 for adresse = 1630 to 1749      <bl>
110 read byte$                      <eh>
120 poke adresse , dec (byte$)      <kb>
130 summe = summe + dec (byte$)     <jb>
140 next                            <jl>
150 if summe <> 9129 then print c4
$"error"                            <ng>
160 print cl$c4$" device error"     <gd>
170 print c4$" initialisierung mit
 sys 1630"                          <dm>
180 print c4$" ausschalten mit
 sys 1641"                          <eb>
190 print c4$" jedes file muss mit
sekundaer-                          <ln>
200 print c4$" adresse eroeffnet w
erden und                           <fc>
210 print c4$" vor dem senden ein
leeres                              <kl>
220 print c4$" print# - kommando g
egeben wer-                         <kp>
230 print c4$" den.                 <dn>
240 print c4$" open 1,4,0 : print#
1                                   <gm>
250 print                           <io>
260 :                               <ee>
270 data a9,72,8d,00,03,a9,06,8d    <em>
280 data 01,03,60,a9,86,8d,00,03    <hi>
290 data 8d,01,03,60,e0,05,f0,03    <pj>
300 data 4c,86,86,a9,86,a0,06,20    <jb>
310 data 88,90,20,e4,ff,f0,fb,60    <bp>
320 data 0d,0d,44,41,53,20,47,45    <ip>
330 data 52,41,45,54,20,49,53,54    <mf>
340 data 20,4e,49,43,48,54,20,41    <fa>
350 data 4e,47,45,53,43,48,41,4c    <ni>
360 data 54,45,54,20,21,0d,4e,41    <kj>
370 data 43,48,20,41,4e,53,43,48    <fg>
380 data 41,4c,54,45,4e,20,42,49    <ga>
390 data 54,54,45,20,54,41,53,54    <fm>
400 data 45,20,44,52,55,45,43,4b    <pl>
410 data 45,4e,0d,00,00,00,00,00    <cf>
420 end                             <ml>
60000 rem nachspann ============= <pj>
60010 rem *farbcodes/steuercodes * <kg>
60020 c4$=chr$(017):cl$=chr$(147)  <ol>
60030 return                       <fo>
```

Rechner steigt aus und die Daten sind verschwunden. Damit das nicht wieder passiert, habe ich diese Routine eingeschaltet, ein Kanal mit Sekundäradresse eröffnet und ein leeres PRINT# gesendet. Wenn kein Fehler auftrat, wird die Routine wieder ausgeschaltet.

PRINT AT

Hat es Sie nicht schon immer gestört, daß mit der CHAR-Anweisung nur fertige Strings übermittelt werden können? Mit dieser Routine wird der Cursor

```
10 rem print at==============c16    <nh>
20 rem (p) commodore welt      ==   <hf>
30 rem ========================     <mm>
40 rem (c) by eckhard schulz   ==   <dc>
50 rem                         ==   <if>
60 rem                         ==   <nd>
70 rem basic 3.5               ==   <mg>
80 rem c16/116/plus4           ==   <fk>
90 rem ========================     <km>
95 gosub 60000                      <jp>
100 for adresse = 1630 to 1666      <bi>
110 read byte$                      <eh>
120 poke adresse , dec (byte$)      <kb>
130 next adresse                    <ba>
200 printcl$c4$" aufruf:            <ed>
210 printc4$" sys 1630,spalte,zeil
e,string                            <hn>
220 printc4$c4$" beispiel: sys1630
,0,1,"chr$(34)"test"chr$(34)        <bc>
230 printc4$" schreibt in zeile 1
und spalte 0 den                    <jl>
240 printc4$" text "chr$(34)"test"
chr$(34)"."                         <cd>
999 :                               <ap>
```

```
1000 data  20,91,94,20,84,9d,8a,c9 <pa>
1010 data  28,b0,17,48,20,91,94,20 <om>
1020 data  84,9d,e0,19,b0,0c,68,a8 <ka>
1030 data  18,20,f0,ff,20,91,94,4c <lf>
1040 data  0b,90,4c,1c,99          <fp>
1050 fort=1630to1666:reada$:poket,
input(a$):next                     <fl>
1060 end                           <na>
60000 rem nachspann ============== <pj>
60010 rem *farbcodes/steuercodes * <kg>
60020 c4$=chr$(017):cl$=chr$(147)  <ol>
60030 return                       <fo>
60040 rem end                      <in>
```

genauso positioniert, aber der Text wird jetzt ausgedruckt wie ein normaler PRINT-Befehl.

SUPER LIST

Mit dieser Routine ist es endlich möglich, ein formatiertes Listing auszugeben. Hier wird nach dem LIST-Befehl jedem Befehl eine Zeile gegeben.
Nun wird auch die längste BASIC-Zeile wieder übersichtlich.

```
10 rem superlist==============c16  <dd>
20 rem (p) commodore welt       == <hf>
30 rem ========================== <mm>
40 rem (c) by eckhard schulz    == <dc>
50 rem                          == <if>
60 rem                          == <nd>
70 rem basic 3.5                == <mg>
80 rem c16/116/plus4            == <fk>
90 rem ========================== <km>
95 gosub 60000                     <jp>
100 for adresse = 1630 to 1709     <bd>
110 read byte$                     <eh>
120 poke adresse , dec (byte$)     <kb>
130 s = s + dec (byte$)            <eo>
140 next adresse                   <ao>
150 if s <> 8910 then printc4$" er
ror":end                           <ae>
160 print c4$" aufruf: sys 1630    <ki>
999 :                              <ap>
1000 data a9,69,8d,06,03,a9,06,8d  <kg>
1010 data 07,03,60,10,03,4c,70,8b  <fb>
1020 data c9,3a,f0,03,4c,4f,8b,a9  <ei>
1030 data 0d,20,d2,ff,98,48,a0,02  <pd>
1040 data b1,5f,aa,c8,b1,5f,85,62  <dg>
1050 data 86,63,a2,90,38,20,ce,a2  <mi>
1060 data 20,71,a4,a2,00,bd,00,01  <ei>
1070 data f0,03,e8,d0,f8,a9,20,20  <dk>
1080 data d2,ff,ca,d0,fa,68,a8,a9  <oc>
1090 data 20,4c,4f,8b,00,00,00,00  <dm>
1100 end                           <ca>
60000 rem nachspann ============== <pj>
60010 rem *farbcodes/steuercodes * <kg>
60020 c4$=chr$(017)                <cj>
60030 return                       <fo>
60040 rem ende                     <ga>
```

BLOCK LOAD

Da es mit dem Monitor nicht möglich ist, an eine bestimmte Adresse zu laden, mußte eine Routine her, die das erlaubt. Das Programm liegt ab Adresse 1630 und benutzt keinerlei BASIC-Speicherplatz. Gestartet wird die Routine mit SYS 1630, gefolgt von ei-

```
10 rem block load=============c16  <ib>
20 rem (p) commodore welt       == <hf>
30 rem ========================== <mm>
40 rem (c) by eckhard schulz    == <dc>
50 rem                          == <if>
60 rem                          == <nd>
70 rem basic 3.5                == <mg>
80 rem c16/116/plus4            == <fk>
90 rem ========================== <km>
95 gosub 60000                     <jp>
100 for adresse = 1630 to 1657     <bh>
110 read byte$                     <eh>
120 poke adresse , dec (byte$)     <kb>
130 summe = summe + dec (byte$)    <jb>
140 next adresse                   <ao>
150 if summe <> 2829 then printc4$
" error":end                       <gi>
160 printcl$c4$" laden von speiche
rbloecken                          <ji>
170 printc4$" an eine bestimmte ad
resse                              <cd>
180 printc4$" aufruf:              <ig>
190 printc4$" sys 1630,adresse,pro
grammname,geraet                   <ci>
200 printc4$" sys 1630,16000,"chr$
(34)"test"chr$(34)",8              <ae>
210 print                          <do>
220 :                              <pd>
230 data 20,de,9d,a5,14,48,a5,15   <ba>
240 data 48,20,91,94,a9,00,85,0a   <mi>
250 data 20,6b,a8,68,a8,68,aa,a9   <ka>
260 data 00,4c,00,a8,00,00,00,00   <kn>
270 end                            <jo>
60000 rem nachspann ============== <pj>
60010 rem *farbcodes/steuercodes * <kg>
60020 c4$=chr$(017):cl$=chr$(147)  <ol>
60030 return                       <fo>
60040 rem end                      <in>
```

nem Komma und einer Adresse, die in dezimaler Form vorliegen muß. Filenamen und Geräteadresse schließen sich an.

DEEK/DOKE

Mit Deek läßt sich der Speicherinhalt zweier nachfolgender Adressen errechnen und einer Variablen zuweisen.
Doke ist das Gegenteil von Deek. Mit diesem Befehl kann man zwei aufeinanderfolgenden Adressen einen neuen Inhalt geben. Um den BASIC-Start nach 6000 hochzulegen, mußte man bisher folgendes eingeben: POKE 43,112:POKE 44,17. Mit dieser Erweiterung nur noch SYS 1667,43,6000

```
10 rem deek/doke=============c16    <mb>
20 rem (p) commodore welt      ==   <hf>
30 rem ========================     <mm>
40 rem (c) by eckhard schulz   ==   <dc>
50 rem                         ==   <if>
60 rem                         ==   <nd>
70 rem basic 3.5               ==   <mg>
80 rem c16/116/plus4           ==   <fk>
90 rem ========================     <km>
95 gosub 60000                      <jp>
100 for adresse = 1630 to 1697      <bp>
110 read byte$                      <eh>
120 poke adresse , dec (byte$)      <kb>
130 summe = summe + dec (byte$)     <jb>
140 next adresse                    <ao>
150 if summe <> 7085 then printc4$
" error":end                        <ge>
160 printcl$c4$" deek und doke fue
r den c16/116  plus/4               <aa>
170 printc4$c4$" deek = sys 1630,a
dresse low,variable                 <ho>
180 printc4$" doke = sys 1667,adre
sse low,wert                        <pb>
190 printc4$c4$" beispiel:          <al>
200 printc4$" sys 1630,43,a         <ab>
210 print" der variablen 'a' wird
der wert der                        <aj>
220 print" adressen 43 und 44 zuge
wiesen.                             <ai>
230 print" das entspricht dem befe
hl:                                 <hh>
240 print" a=peek(43)+peek(44)*256  <ki>
250 printc4$" sys 1667,43,6000      <jg>
260 print" hier wird in die adress
en 43 und 44 der                    <gg>
270 print" wert 6000 gepoked.       <ge>
280 print" das entspricht dem befe
hl:                                 <fp>
290 print" poke 43,112:poke44,17    <gn>
300 :                               <je>
310 data 20,99,06,a2,00,a1,14,85    <kg>
320 data 63,e6,14,a1,14,85,62,a2    <bb>
330 data 90,38,20,ce,a2,20,91,94    <eo>
340 data 20,a5,96,85,49,84,4a,20    <cf>
350 data 1b,93,4c,55,a2,20,99,06    <nk>
360 data 84,71,85,72,20,99,06,a2    <ci>
370 data 00,98,81,71,e6,71,a5,15    <eh>
380 data 81,71,60,20,91,94,20,14    <nm>
390 data 93,4c,e4,9d,00,00,00,00    <ak>
400 end                             <kd>
60000 rem nachspann =============   <pj>
60010 rem *farbcodes/steuercodes *  <kg>
60020 c4$=chr$(017):cl$=chr$(147)   <ol>
60030 return                        <fo>
60040 rem ende                      <ga>
```

DEEK entspricht einem erweiterten PEEK und DOKE einem erweiterten POKE.

Eckhard Schulz □

DIAGRAMM

Optik für Ihre Zahlen

Übersichtlich und anschaulich präsentieren sich sonst nichtssagende Zahlen, wenn sie in Form von Balken-, Kreis- und X-Y-Diagrammen für das Auge aufbereitet werden. Keine besonderen Anstrengungen sind für die grafische Ausarbeitung vonnöten, sondern nur das vorliegende Programm.

Das Programm Diagramm 16 bietet die Möglichkeit, Zahlenangaben (zum Beispiel Statistiken) in anschaulichen und übersichtlichen Grafiken darzustellen.
Prinzipiell stehen drei verschiedene Diagramme zur Auswahl, deren Verwendung durch Art und Umfang der darzustellenden Werte bestimmt wird.
Nach dem Start des Programmes wird zunächst eine Maschinenroutine zum Laden und Speichern von Hires-Bildern eingelesen und im Kassettenpuffer abgelegt.
Danach gelangt man in das Hauptmenü mit den Punkten:
1. Demos
2. Balkendiagramm
3. Kreisdiagramm
4. x-y-Diagramm
5. Laden/Speichern
6. Programmende

1. Demos

Man gelangt in ein Untermenü, aus dem heraus zu jedem der drei möglichen Diagramme eine Demo abgerufen werden kann. Hierdurch gewinnt man am schnellsten einen Überblick über die Einsatzmöglichkeiten der Grafiken. Durch Druck einer Taste wird anschließend wieder das Hauptmenü angesprungen.

2. Balkendiagramm

Nach einer kurzen Einleitung werden folgende Eingaben verlangt:
- Überschrift der Grafik (maximal 40 Zeichen lang);
- Maßeinheit der Zahlenwerte;
- Anzahl der darzustellenden Balken (1 bis 6);
- Zahlenwerte (1 bis 6) und deren Bezeichnung.

Anschließend wird die Grafik aufgebaut. Die Balken werden räumlich umrissen und farblich unterschieden. Überschrift, Maßeinheit, Zahlenwerte und Bezeichnung werden durch CHAR-Befehle plaziert. Durch Druck einer Taste gelangt man wieder in das Hauptmenü.

3. Kreisdiagramm

Das Kreisdiagramm stellt Prozentzahlen als Segmente eines aufgesprengten Vollkreises dar.
Auf eine kurze Einleitung folgt die Eingabe der unter 2. beschriebenen Punkte, wobei die Angabe der Maßeinheit entfällt. Es können bis zu sechs Prozentzah-

len und deren Bezeichnungen eingegeben werden. Die Summe der Prozentangaben darf 100 Prozent nicht überschreiten, sonst muß noch einmal eingegeben werden. Angaben kleiner als 5 Prozent sollten vermieden werden, da sonst Fehler in der Grafik auftreten können. Nach Abschluß der Eingaben wird das Bild erstellt. Die Segmente werden umrissen und ausgefüllt. Die Prozentangaben erscheinen neben dem jeweiligen Kreisausschnitt, die Bezeichnungen in den unteren Bildschirmzeilen. Auch hier wird nach dem Drükken einer Taste wieder in das Hauptmenü gesprungen.

4. x-y-Diagramm

Mit diesem Diagramm kann der Verlauf eines bestimmten Zahlenwertes über der Zeit (ein Jahr) aufgezeigt werden. Die x-Achse ist die Zeitachse. Sie umfaßt ein Jahr, unterteilt in die zwölf Monate. Die y-Achse kann mit einer beliebigen Maßeinheit belegt werden. Die einzelnen Zahlenwerte dürfen nicht mehr als fünf Stellen umfassen.
Nach einer kurzen Einleitung erfolgen die bereits beschriebenen Eingaben, wobei den einzelnen Werten hier keine Bezeichnung mehr zugeordnet wird.
Zunächst werden die Achsen gezeichnet und beschriftet. Die y-Achse ist immer in sechs Abschnitte unterteilt. Anhand des maximalen eingegebenen Zahlenwertes wird die Achsenbeschriftung berechnet, wobei geringfügige Rundungsfehler auftreten. Werte mit mehr als drei Stellen werden durch Division „verkürzt". Der entsprechende Faktor für die Maßeinheit wird mit angezeigt.
Anschließend wird die Kurve als Verbindungslinie von Wert zu Wert gezeichnet. Die Kurve wird zur Verdeutlichung in dreifacher Breite erstellt. Ein Tastendruck führt zurück in das Hauptmenü.

5. Laden/Speichern

Es wird das folgende Untermenü angesprungen:
1. Directory
2. Speichern
3. Laden
4. Hauptmenü

Punkt 1. zeigt lediglich den Inhalt der eingelegten Diskette, um die Suche nach bestimmten Grafiken zu erleichtern.
Mit Punkt 2. kann das Diagramm im Grafikspeicher auf Diskette abgespeichert werden. Dazu wird zunächst die vorhandene Grafik kurz eingeblendet. Es erfolgt die Abfrage, ob gespeichert werden soll. J führt zur Abfrage des Filenamens, alle anderen Tasten bewirken den Rücksprung in das Untermenü. Der Filename darf bis zu 15 Zeichen lang sein. Durch einen SYS-Befehl wird die Maschinenroutine zum Speichern aufgerufen. Es wird der Bereich von 1800 bis 3FFF (hex) gespeichert, damit auch die Helligkeits- und Farbinformationen erhalten bleiben. Nach Beendigung des Speichervorganges springt das Programm zurück in das Untermenü.
Mit Punkt 3. können auf diese Weise abgespeicherte Grafiken wieder eingeladen werden. Zuvor wird das Directory zur Auswahl des Filenamens angezeigt. Während des Ladevorganges ist der Bildschirm abgeschaltet. Wer den Aufbau der Grafik verfolgen will, muß die beiden POKE-Befehle in Zeile 2360 entfernen. Die eingelesene Grafik kann nun betrachtet werden. Der Rücksprung zum Untermenü erfolgt durch Druck einer Taste.
Puntk 4. bewirkt den Rücksprung in das Hauptmenü.

Bitte lesen Sie weiter auf Seite 46

```
10 rem diagramm=================p4 <pl>
20 rem (p) commodore welt team     <ho>
30 rem ===========================  <ng>
40 rem (c) by henning koglin       <ia>
50 rem                             <pd>
60 rem basic v3.5                  <od>
70 rem plus4 (c16/116 + 64 kb)     <cd>
80 rem floppy                      <ak>
90 rem ===========================  <jg>
100 gosub2390:rem maschinencode    <nh>
110 rem -----------                <ip>
120 rem hauptmenue                 <po>
130 rem ----------                 <lj>
140 scnclr:clr:gosub2550:color0,1:
color4,1:color1,7,5                <no>
150 dimp(12):dimpr(12)             <en>
160 char1,11,0,rn$+"d i a g r a m
m e"+rf$                           <be>
170 char1,11,3,"1.demos.........."
:char1,11,5,"2.balkendiagramm."    <od>
180 char1,11,7,"3.kreisdiagramm.."
:char1,11,9,"4.x-y-diagramm..."    <jn>
190 char1,11,11,"5.laden/speichern
":char1,11,13,"6.programmende..."  <eh>
200 char1,7,17,"bitte waehlen sie
(1-6) !"                           <ei>
210 geta$:ifa$>"0"anda$<"7"then220
:else210                           <fk>
220 onval(a$)goto260,550,830,1250,
1960,1670                          <km>
230 rem ---------------            <hl>
240 rem demo-untermenue            <ik>
250 rem ---------------            <ff>
260 scnclr:char1,11,0,rn$+b3$+"d e
 m o s"+b4$+rf$:char1,15,2,b3$+"zu
"                                  <fi>
270 char1,11,4,"1.balkendiagramm":
char1,11,6,"2.kreisdiagramm."      <ji>
280 char1,11,8,"3.x-y-diagramm..":
char1,7,15,"bitte waehlen sie (1-3
) !"                               <gp>
290 geta$:ifa$>"0"anda$<"4"then300
:else290                           <dk>
300 onval(a$)goto340,380,450       <dh>
310 rem ------------------         <mg>
320 rem demo balkendiagramm        <gn>
330 rem ------------------         <ob>
340 c=6:d$="ausgaben im monat deze
mber":me$="dm"                     <ho>
350 p$(1)="miete":p(1)=825.30:p$(2
)="strom":p(2)=70.00:p$(5)="auto":
p$(6)="hobby"                      <mc>
360 p$(3)="haushalt":p(3)=552.28:p
$(4)="kinder":p(4)=239.10:p(5)=83.
20:p(6)=120                        <hl>
370 goto690                        <pc>
380 rem ----------------           <kn>
390 rem demo kreisdiagramm         <ih>
```

```
400 rem -----------------                   <fc>
410 c=6:d$="ausgaben im 1.halbjahr
 1988"                                      <da>
420 p$(1)="kleidung":p(1)=6.8:p$(2
)="auto":p(2)=14.4:p$(3)="moebelka
uf":p(3)=21.5                               <pc>
430 p$(4)="hobby":p(4)=15.3:p$(5)=
"haushalt":p(5)=22.5:p$(6)="sonsti
ges":p(6)=19.5                              <gm>
440 goto990                                 <gl>
450 rem -----------------                   <cg>
460 rem demo x-y-diagramm                   <gp>
470 rem -----------------                   <kj>
480 c=12:d$="krankheitskosten 1988
":me$="dm"                                  <hl>
490 p(1)=76:p(2)=54:p(3)=20:p(4)=2
1:p(5)=18:p(6)=5                            <hb>
500 p(7)=66:p(8)=19:p(9)=12:p(10)=
7:p(11)=19:p(12)=32                         <ne>
510 goto1430                                <hj>
520 rem --------------                      <gg>
530 rem balkendiagramm                      <da>
540 rem --------------                      <al>
550 scnclr:printrn$"b a l k e n d
i a g r a m m"rf$                           <hp>
560 printc4$"das balkendiagramm st
ellt bis zu sechs"                          <im>
570 print"verschiedene zahlenwerte
 grafisch dar."                             <bp>
580 print"die eingabe kann in beli
ebiger massein- "                           <jb>
590 print"heit erfolgen."                   <dj>
600 printc4$c4$c4$"bitte ueberschr
ift der grafik eingeben !"                  <bj>
610 print"(nicht laenger als 40 ze
ichen) !":inputd$                           <pn>
620 iflen(d$)>40then550                     <ki>
630 printc4$c4$"in welcher massein
heit soll dargestellt":print"werde
n";:inputme$                                <jm>
640 char1,0,18,"wieviele balken (1
-6)":inputc$                                <nk>
650 ifc$<"7"andc$>"0"then660:else6
40                                          <dp>
660 c=val(c$):scnclr                        <ae>
670 fori=1toc:print"eingabe ";i;".
 wert"b3$b4$;:inputp(i)                     <ld>
680 print"fuer (bezeichnung)"b3$b3
$;:inputp$(i):nexti                         <io>
690 gosub1720                               <ga>
700 graphic1,1                              <ci>
710 fori=1toc:color1,2+i,4                  <am>
720 box1,i*48-48,168-pr(i),i*48-24
,168                                        <mb>
730 draw1,i*48-24,168toi*48-12,160
toi*48-12,160-pr(i)toi*48-24,168-p
r(i)                                        <co>
740 draw1,i*48-24,168-pr(i)toi*48-
12,160-pr(i)toi*48-36,160-pr(i)toi
*48-48,168-pr(i)                            <pc>
750 paint1,i*48-45,167,1                    <bh>
760 nexti:color1,2:z=1                      <mo>
770 fori=1toc:char1,(i-1)*6,5+z,st
r$(p(i)):char1,(i-1)*6,23+z,p$(i):
z=-z:nexti                                  <co>
780 gosub1810:gosub1870                     <bc>
790 getkeyw$:graphic0:goto140               <ci>
800 rem -------------                       <cp>
810 rem kreisdiagramm                       <hd>
820 rem -------------                       <oc>
830 scnclr:char1,0,0,rn$+"k r e i
s d i a g r a m m "+rf$:print               <ia>
840 printc4$c4$"das kreisdiagramm
stellt bis zu sechs"                        <bg>
850 print"prozentwerte als teile e
ines vollkreises"                           <li>
860 print"dar. die summe der werte
 darf 100% nicht"                           <df>
870 print"ueberschreiten !"                 <ek>
880 print"eingaben unter 5% sollte
n vermieden":print"werden !!!"              <eh>
890 char1,0,10,"bitte ueberschrift
 der grafik eingeben !"                     <lb>
900 char1,0,11,"(nicht laenger als
 40 zeichen) !":print:inputd$               <hf>
910 iflen(d$)>40then830                     <dp>
920 char1,0,14,"wieviele prozentwe
rte (1-6)":inputc$                          <nb>
930 ifc$>"0"andc$<"7"then940:else9
20                                          <mi>
940 c=val(c$):scnclr                        <ie>
950 fori=1toc:print"eingabe ";i;".
 prozentwert ";:inputp(i)                   <ea>
960 print"fuer (bezeichnung)"b3$b4
$;:inputp$(i):nexti:su=0                    <lf>
970 fori=1toc:su=su+p(i):nexti              <ag>
980 ifsu>100then940                         <pi>
990 fori=1toc:pr(i)=int((p(i)*3.6)
+.5):nexti                                  <di>
1000 graphic1,1:color1,8,5                  <al>
1010 r=60:su=0:x=160:y=100:locatex
,y                                          <ef>
1020 fori=1toc                              <ib>
1030 locate9;(su+pr(i)/2):xx=rdot(
0):yy=rdot(1):circle1,xx,yy,r,r,su
,su+pr(i)                                   <ip>
1040 drawtoxx,yy:drawtor;su:su=su+
pr(i)                                       <lm>
1050 locatex,y:locate30;(su-pr(i)/
2):paint                                    <pj>
1060 locatex,y:nexti                        <fl>
1070 color1,2:gosub1810                     <ih>
1080 su=0                                   <eh>
1090 fori=1toc:locatex,y:locater+2
5;(su+pr(i)/2)                              <di>
1100 xz=int(rdot(0)/8):yz=int(rdot
```

```
(1)/8):su=su+pr(i):ifyz<2thenyz=2  <pk>
1110 ifyz>21thenyz=21  <nc>
1120 ifyz<21andxz<19thenxz=xz-len(
str$(p(i)))-3  <on>
1130 color1,3,6  <eg>
1140 char1,xz,yz,str$(i)+":"+str$(
p(i))+"%":nexti  <gh>
1150 color1,2:fori=0to39:char1,i,2
2,"C":nexti  <co>
1160 color1,7,6  <np>
1170 ifc>3thenfori=1to3:elsefori=1
toc  <jk>
1180 char1,(i-1)*13,23,str$(i)+":"
+p$(i):nexti  <dp>
1190 ifc>3thenfori=4toc:else1210  <kj>
1200 char1,(i-4)*13,24,str$(i)+":"
+p$(i):nexti  <ef>
1210 getkeyw$:graphic0:goto140  <kh>
1220 rem ------------  <gb>
1230 rem x-y-diagramm  <ok>
1240 rem ------------  <oi>
1250 scnclr:char1,0,0,rn$+"x - y -
 d i a g r a m m "+rf$  <bo>
1260 print:printc4$c4$"das x-y-dia
gramm stellt bis zu zwoelf"  <ib>
1270 print"zahlenwerte dar. die y-
achse ist die"  <ol>
1280 print"zeitachse, die in die z
woelf monate"  <gp>
1290 print"unterteilt ist. "  <bg>
1300 print"die x-achse kann mit ei
ner beliebigen"  <lm>
1310 print"masseinheit belegt werd
en. sie wird"  <ej>
1320 print"grundsaetzlich in sechs
 abschnitte auf-"  <on>
1330 print"geteilt, wobei der maxi
malwert zwischen"  <co>
1340 print"0 und 99999 liegen kann
."  <fm>
1350 char1,0,14,"bitte ueberschrif
t der grafik eingeben !"  <nb>
1360 char1,0,15,"(nicht laenger al
s 40 zeichen) !":print:inputd$  <jh>
1370 iflen(d$)>40then1250  <eh>
1380 printc4$c4$"in welcher massei
nheit soll dargestellt":print"werd
en";:inputme$  <jf>
1390 char1,0,23,"wieviele werte (2
-12)":inputc$  <ki>
1400 ifval(c$)>1andval(c$)<13then1
410:else1390  <ma>
1410 c=val(c$):scnclr  <jh>
1420 fori=1toc:print"eingabe ";i;"
. wert";:inputp(i):print:nexti  <ic>
1430 gosub1720  <eh>
1440 z=1:ux=40:uy=160:restore1950  <ki>
1450 graphic1,1:locateux,uy:color1
,8,5:drawto320,uy:locateux,160:dra
wtoux,56  <fn>
1460 fori=1to12:locate28+i*24,160:
drawto2;0to4;180  <km>
1470 reada$:char1,((rdot(0)-24)/8)
+1,22+z,a$:z=-z:nexti  <ae>
1480 fori=1to6:locateux,uy-(i*16):
drawto2;270to4;90:nexti  <kp>
1490 pm=pm*100  <bl>
1500 ifpm<1000andpm>0thenmu=1  <bp>
1510 ifpm<10000andpm>=1000thenmu=.
1:me$=me$+" * 10"  <cc>
1520 ifpm>=10000thenmu=.01:me$=me$
+" * 100"  <fc>
1530 ps=int(pm*mu/6)  <dk>
1540 fori=1to6:pa=(7-i)*ps:char1,0
,i*2+6,str$(pa)  <hf>
1550 nexti  <jo>
1560 color1,3,5  <ck>
1570 gosub1600:r=1:gosub1640:gosub
1600:r=-2:gosub1640:gosub1600  <bh>
1580 color1,2:gosub1810:gosub1870  <pg>
1590 getkeyw$:graphic0:goto140  <no>
1600 locate52,160-pr(1)  <dk>
1610 fori=2toc  <kb>
1620 xp=(28+i*24):drawtoxp,160-pr(
i):nexti  <kn>
1630 return  <fg>
1640 fori=1toc:pr(i)=pr(i)-r  <oa>
1650 ifpr(i)<0thenpr(i)=0  <ej>
1660 nexti:return  <gd>
1670 rem ------------  <ck>
1680 rem programmende  <bd>
1690 rem ------------  <lc>
1700 scnclr:color1,8,7:char1,2,12,
"programm wirklich beenden (j/n) ?
?"  <gk>
1710 getkeyw$:ifw$="j"thensys65529
:else140  <dn>
1720 rem ------------------------  <gg>
1730 rem maximalwert aussortieren  <pi>
1740 rem ------------------------  <ck>
1750 pm=0  <ba>
1760 fori=1toc:ifp(i)>pmthenpm=p(i
)  <ma>
1770 nexti  <ap>
1780 pm=pm/100  <nb>
1790 fori=1toc:pr(i)=int((p(i)/pm)
+.5):nexti  <pd>
1800 return  <km>
1810 rem -------------------------
-  <io>
1820 rem ueberschrift positioniere
n  <of>
1830 rem -------------------------
-  <jl>
1840 d=int((40-len(d$))/2):char1,d
,0,d$  <mf>
```

```
1850 fori=0to39:char1,i,1,"C":next
i                                  <ib>
1860 return                        <ce>
1870 rem ------------------------- <db>
1880 rem masseinheit positionieren <ld>
1890 rem ------------------------- <jm>
1900 ma$="angaben in "+me$:me=int(
(40-len(ma$))/2):char1,me,2,ma$    <ko>
1910 return                        <ii>
1920 rem -----                     <ee>
1930 rem datas                     <mj>
1940 rem -----                     <ep>
1950 data jan,feb,mae,apr,mai,jun,
jul,aug,sep,okt,nov,dez            <ag>
1960 rem -------------------------
-                                  <be>
1970 rem untermenue laden/speicher
n                                  <dc>
1980 rem -------------------------
-                                  <ni>
1990 scnclr:char1,12,0,rn$+"laden/
speichern"+rf$                     <do>
2000 char1,12,3,"1.directory....":
char1,12,5,"2.speichern...."       <ah>
2010 char1,12,7,"3.laden........":
char1,12,9,"4.hauptmenue..."       <dd>
2020 char1,7,17,"bitte waehlen sie
 (1-4) !"                          <kl>
2030 geta$:ifa$>"0"anda$<"5"then20
40:else2030                        <bf>
2040 onval(a$)goto2050,2090,2220,1
40                                 <ac>
2050 rem ---------                 <op>
2060 rem directory                 <kf>
2070 rem ---------                 <lm>
2080 scnclr:directory:printc4$"wei
ter mit beliebiger taste !":getkey
w$:goto1960                        <ji>
2090 rem ---------                 <ap>
2100 rem speichern                 <nj>
2110 rem ---------                 <cc>
2120 scnclr:print"zur zeit befinde
t sich die folgende":print"grafik
im speicher:"                      <cn>
2130 fori=1to2000:nexti            <ec>
2140 graphic1:fori=1to3000:nexti:g
raphic0                            <ok>
2150 printc4$c4$c4$"speichern (j/n
) ?";:getkeyw$:print" "+w$         <le>
2160 ifw$="j"then2170:else1960     <jk>
2170 char1,0,12,"name der grafik "
:inputna$                          <ln>
2180 iflen(na$)>15then2170         <jh>
2190 iflen(na$)<15thenfort=len(na$
)to15:na$=na$+" ":nextt            <jf>
2200 fori=1015to1030:pokei,asc(mid
$(na$,i-1014,1)):nexti             <bi>
2210 sys818:goto1960               <fo>
2220 rem -----                     <if>
2230 rem laden                     <db>
2240 rem -----                     <he>
2250 scnclr:print"mit dieser routi
ne koennen sie grafiken"           <eh>
2260 print"laden, die auf der eing
elegten diskette"                  <oc>
2270 print"gespeichert sind.":prin
t"zunaechst erscheint die director
y."                                <pm>
2280 print"anschliessend ist der n
ame der ge-":print"wuenschten graf
ik einzugeben."                    <pk>
2290 print"danach wird die grafik
eingelesen. sie"                   <pl>
2300 print"kann nun betrachtet wer
den."                              <im>
2310 print"durch druck einer taste
 gelangt man wie-"                 <gb>
2320 print"der in das untermenue '
laden/speichern'."                 <gl>
2330 char1,09,24,"bitte taste drue
cken !":getkeyw$                   <og>
2340 scnclr:directory:print:input"
name der grafik";na$               <mf>
2350 iflen(na$)>15then2340         <ok>
2360 iflen(na$)<15thenfort=len(na$
)to15:na$=na$+" ":nextt            <kf>
2370 fori=1015to1030:pokei,asc(mid
$(na$,i-1014,1)):nexti             <fn>
2380 graphic1:poke65286,11:sys854:
poke65286,27:getkeyw$:graphic0:got
o1960                              <ch>
2390 rem ------------------------- <kl>
2400 rem ms-unterprogramm einlesen <ni>
2410 rem ------------------------- <gl>
2420 restore2480                   <gl>
2430 fori=818to877                 <jo>
2440 readms$                       <ff>
2450 pokei,dec(ms$)                <ee>
2460 nexti                         <oe>
2470 return                        <on>
2480 data a9,01,a2,08,a0,08,20,ba,
ff,a9                              <bf>
2490 data 0f,a2,f7,a0,03,20,bd,ff,
a2,00                              <ek>
2500 data a0,18,86,d8,84,d9,a9,d8,
a2,ff                              <hm>
2510 data a0,3f,20,d8,ff,60        <ac>
2520 data a9,01,a2,08,a0,01,20,ba,
ff,a9                              <hf>
2530 data 0f,a2,f7,a0,03,20,bd,ff,
a9,00                              <hb>
2540 data 20,d5,ff,60              <jf>
2550 rem nachspann ============== <hl>
2560 rem * farbcodes/steuercodes * <ao>
2570 c4$=chr$(017):rn$=chr$(018)   <kd>
2580 rf$=chr$(146):b$=chr$(32)     <eh>
```

```
2590 b3$=b$+b$+b$:b4$=b3$+b$           <gg>
2600 return                            <pc>
2610 rem diagramm===============p4     <nd>
2620 rem 60671 bytes memory            <la>
2630 rem 08408 bytes program           <jk>
2640 rem 00049 bytes variables         <be>
2650 rem 00144 bytes arrays            <fb>
2660 rem 00338 bytes strings           <li>
2670 rem 12288 bytes graphic           <oh>
2680 rem 39444 bytes free (0)          <jc>
2690 rem ========================      <jd>
```

DIAGRAMM

Fortsetzung von Seite 42

6. Programmende

Nach einer Sicherheitsabfrage wird das Programm durch einen Software-Reset gelöscht. Der Grafikspeicher bleibt erhalten.

Unter Verwendung eines Druckers und einer Hardcopy-Routine (wie die Hires-Hardcopy aus C16-P4 SPECIAL Nr. 1) lassen sich die erstellten Diagramme natürlich auch zu Papier bringen.

Henning Koglin □

INTER-COPY

Von Kassette auf Diskette

Wer sich eine Diskettenstation angeschafft hat, würde seine Software, die sich noch auf Kassetten befindet, gerne auf das neue Speichermedium kopieren. Autostart und Kopierschutz vereiteln jedoch dieses Vorhaben. In den meisten Fällen kann Inter-Copy helfen.

Wenn das zu kopierende Programm sich mit dem Speicherplatz eines nicht erweiterten C16 begnügt, nach dem Laden den Interrupt wieder zuläßt und beim Laden weder den Interruptvektor verbiegt noch das ROM einschaltet, steht dem Kopieren von Kassette auf Diskette nichts mehr im Wege.
Das Programm Inter-Copy wartet darauf, daß nach dem Ladevorgang der Interrupt wieder zugelassen wird. Sobald das geschehen ist, werden sofort innerhalb eines Interruptes der ganze Bereich von $0000 bis $0FFF sowie die Ted-Register in einen anderen Speicherbereich verschoben und ein Reset ausgeführt. Das Programm startet also nicht automatisch. Um ein so abgebrochenes Programm wieder zum Laufen zu bringen, wird nicht die Startadresse gebraucht, sondern der Stackpointer. Man muß also außer dem ei-

Bitte lesen Sie weiter auf Seite 48

```
10 rem intercopy================p4 <oj>
20 rem (p) commodore welt team     <ho>
30 rem ========================== <ng>
40 rem (c) by martin schulz        <me>
50 rem                             <pd>
60 rem basic v3.5                  <od>
70 rem plus4 (c16/116 + 64kb)      <cd>
80 rem floppy und datasette        <hh>
90 rem ========================== <jg>
100 fori=312to348                  <bi>
110 reada:pokei,a:next             <ne>
120 data 201,013,208,003,076,075   <oc>
130 data 236,134,208,041,015,170   <im>
140 data 165,209,072,165,210,133   <ko>
150 data 209,104,133,210,032,075   <co>
160 data 236,202,208,250,166,208   <ba>
170 data 024,096,024,096,076,075   <dm>
180 data 236                       <mi>
190 color4,6,4:color0,7,6          <ia>
200 scnclr:print:print:print       <fm>
210 poke209,32:poke210,166         <ei>
220 poke804,56:poke805,1           <mk>
230 print"adcbbbbfbfbe"            <bf>
240 print"bbdcabdbdbfbbb"          <ah>
250 print"bbdfdbdbfbbb"            <bh>
260 print"bbdfdbddde"              <hg>
270 print"bbdbacdbdbfd"            <eh>
280 print"bbdbbbdbdbfbab"          <di>
290 print"adcbbbdbdfbbbb"          <dm>
300 print:print:print              <hm>
310 print"ddddcecbbb"              <gg>
320 print"cbbbbbbbbbbbbbbb"        <lk>
330 print"cbfbbbbbbbbbbb"          <dh>
340 print"cbfbbbbedd"              <np>
350 print"cbfbbbbbhb"              <bd>
360 print"cbbbbbbbbbhb"            <al>
370 print"ddddcbhb"                <cm>
380 poke804,75:poke805,236         <cl>
390 print:print:print:print        <fa>
400 print"      druecken sie die sp
ace-taste !"                       <kh>
410 getkey a$:print:print:print    <gg>
420 print"  inter-copy ist ein pro
gramm, das es":print               <pc>
430 print"  ermoeglicht, c16-progr
amme mit auto-":print              <di>
440 print"  start von kassette auf
 diskette zu":print                <ha>
450 print"  uebertragen. hierzu er
forderlich ist":print              <nn>
460 print"  ein c16 mit 64 kb.-erw
eiterung oder":print               <gc>
470 print"  ein plus 4. da es vers
chiedene":print                    <dn>
480 print"  moeglichkeiten des kop
ierschutzes":print                 <ob>
490 print"  gibt, gelingt diese ar
t des kopierens":print             <cg>
```

```
500 print"  nicht bei jedem progra
mm. auch dort,":print                 <hc>
510 print"  wo programmteile von k
assette nachge-":print                <pd>
520 print"  laden werden, ist inte
r-copy nur":print                     <mj>
530 getkey a$                         <ni>
540 print"  bedingt anwendbar. der
 vorteil bei":print                   <cp>
550 print"  inter-copy liegt darin
, dass die":print                     <lk>
560 print"  startadresse eines pro
grammes nicht":print                  <do>
570 print"  mehr gebraucht wird. w
enn sie gleich":print                 <mb>
580 print"  die space-taste drueck
en, wird der":print                   <fl>
590 print"  bildschirm geloescht u
nd die ein-":print                    <be>
600 print"  schaltmeldung erschein
t:":print                             <lp>
610 print"      inter-copy      1227
7 bytes free":print                   <fg>
620 print"  inter-copy ist dann ak
tiviert.":print                       <fj>
630 print"  bevor sie nun ein prog
ramm kopieren,":print                 <el>
640 print"  laden sie es einmal zu
r probe und":print                    <db>
650 getkey a$                         <fo>
660 print"  merken sie sich den st
and des band-":print                  <cd>
670 print"  zaehlwerkes, nachdem e
s geladen ist!":print                 <ek>
680 print"  nun koennen sie die fo
lgenden":print                        <bd>
690 print"  programmschritte zum k
opieren eines":print                  <jj>
700 print"  programmes anwenden. b
evor sie das":print                   <bi>
710 print"  tun, notieren sie sich
 diese bitte !":print                 <ce>
720 print                             <do>
730 print"1.  laden sie inter-copy
 und starten es.":getkey a$:print     <di>
740 print"2.     laden sie das zu k
opierende"                            <eg>
750 print"       programm von kasse
tte.":print                           <on>
760 print"3.     kurz bevor der lad
evorgang"                             <ge>
770 print"  abgeschlossen ist, (1-
2 nummern des"                        <lp>
780 print"  bandzaehlwerkes vorher
), druecken sie"                      <kh>
790 print"  die tasten shift und a
 gleichzeitig"                        <jl>
800 print"  und halten diese gedru
eckt, bis der"                        <oo>
810 print"  ladevorgang beendet is
t.":print                             <hg>
820 print"4.  geben sie nun den be
fehl  monitor"                        <eh>
830 print"  ein und druecken sie d
ie return-taste."                     <ln>
840 print"  speichern sie nun das
programm mit"                         <kg>
850 print"  s"chr$(34)"programmnam
e"chr$(34)",8,1000,5060"              <lb>
860 print"  auf diskette ab.":prin
t                                     <pa>
870 print"5.     schalten sie nun d
en computer aus"                      <dl>
880 print"  und wieder an. geben s
ie den befehl"                        <ke>
890 print"  monitor   ein und drue
cken die return-"                     <dm>
900 print"  taste. laden sie das s
oeben abge-"                          <fj>
910 print"  speicherte programm mi
t"                                    <pb>
920 print"  l"chr$(34)" programmna
me"chr$(34)",8   und starten sie"     <ma>
930 print"  es mit      g 5000"       <hm>
940 getkey a$:scnclr:char,13,4,"mo
ment bitte !"                         <ec>
950 char,7,6,"inter-copy wird akti
viert !"                              <hk>
960 for i=0 to 433                    <bg>
970 read x:poke 20480+i,x             <hm>
980 next:sys dec("503c")              <lc>
990 data 120,160,0,185,0,64,153,0     <ej>
1000 data 0,200,208,247,238,5,80,2
38                                    <ng>
1010 data 8,80,173,8,80,201,16,208    <ho>
1020 data 234,185,64,80,153,0,255,
200                                   <cg>
1030 data 192,32,208,245,173,60,80
,141                                  <hf>
1040 data 0,16,173,61,80,141,1,16     <dc>
1050 data 173,62,80,141,2,16,162,2
48                                    <cn>
1060 data 154,76,195,252,120,160,0
,185                                  <no>
1070 data 0,128,153,0,128,200,208,
247                                   <pd>
1080 data 238,65,80,238,68,80,173,
68                                    <mj>
1090 data 80,201,253,208,234,160,1
92,185                                <cj>
1100 data 63,255,153,63,255,136,20
8,247                                 <gh>
1110 data 169,63,32,180,80,141,63,
255                                   <fa>
1120 data 169,76,141,14,206,169,23
3,141                                 <hc>
```

```
1130 data 15,206,169,80,141,16,206
,160                              <gn>
1140 data 10,185,245,255,153,245,6
3,136                             <do>
1150 data 208,247,76,143,81,162,12
6,232                             <pk>
1160 data 240,24,189,0,68,201,177,
208                               <mo>
1170 data 246,232,232,232,16,12,18
9,0                               <mj>
1180 data 68,201,63,208,234,222,0,
68                                <ea>
1190 data 208,229,173,222,71,201,1
77,208                            <eo>
1200 data 10,173,225,71,201,63,208
,3                                <pl>
1210 data 206,225,71,96,141,50,129
,141                              <jf>
1220 data 78,129,141,187,207,141,2
47,255                            <kl>
1230 data 141,130,4,141,158,4,141,
172                               <jk>
1240 data 4,141,183,4,141,194,4,14
1                                 <ij>
1250 data 205,4,141,216,4,141,227,
4                                 <hi>
1260 data 141,225,7,96,141,63,255,
76                                <bl>
1270 data 66,206,141,63,255,76,14,
206                               <jn>
1280 data 3,169,0,141,48,253,141,8 <ed>
1290 data 255,173,8,255,201,123,20
8,111                             <ef>
1300 data 173,232,80,240,106,186,1
42,55                             <mg>
1310 data 80,160,0,185,0,0,153,0  <mf>
1320 data 64,200,208,247,238,5,81,
238                               <ef>
1330 data 8,81,173,5,81,201,16,208 <mp>
1340 data 234,185,0,255,153,64,80,
200                               <kd>
1350 data 192,32,208,245,169,0,141
,232                              <of>
1360 data 80,169,62,32,180,80,169,
206                               <lo>
1370 data 141,19,3,141,21,3,141,19 <md>
1380 data 67,141,21,67,169,66,141,
18                                <ao>
1390 data 3,141,18,67,169,14,141,2
0                                 <ho>
1400 data 3,141,20,67,173,0,16,141 <md>
1410 data 60,80,173,1,16,141,61,80 <po>
1420 data 173,2,16,141,62,80,32,13
3                                 <ij>
1430 data 80,141,62,255,76,164,242
,173                              <ae>
1440 data 19,3,201,206,208,10,169,
80                                <lo>
1450 data 141,19,3,169,220,141,18,
3                                 <ik>
1460 data 173,21,3,201,206,208,10,
169                               <gk>
1470 data 80,141,21,3,169,226,141,
20                                <je>
1480 data 3,173,9,255,76,17,206,16
0                                 <na>
1490 data 20,185,157,81,153,207,12
8,136                             <ci>
1500 data 16,247,76,164,242,32,32,
32                                <mj>
1510 data 32,73,78,84,69,82,45,67 <mf>
1520 data 79,80,89,32,32,32,32,32,
32,32                             <fk>
1530 rem ======================== <ge>
1540 rem  p r o g r a m m e n d e <ie>
1550 rem ======================== <op>
```

Inter-Copy

Fortsetzung von Seite 46

gentlichen Programm den kopierten Bereich $0000 bis $0FFF (jetzt $4000 bis $4FFF) und das Startprogramm mit dem Stackpointer auf Diskette abspeichern.
Man kann das Programm nun von Diskette laden und vom Monitor aus mit G5000 starten. Als Beweis, daß dies funktioniert, habe ich das Programm „Atlantis" auf diese Weise auf Diskette abgespeichert.
Es ist durch einen Reset nicht zu stoppen. Da ich die Interruptroutine erweitern mußte, habe ich das ganze ROM ins RAM kopiert. Die Erklärung zu Inter-Copy habe ich mit ins Programm eingebaut. Das eigentliche Programm beginnt mit Zeile 960.

Zur Anwendung:

1. Inter-Copy laden und starten. Programm von Kassette laden.
2. Kurz vor Abschluß des Ladevorganges die Tasten Shift und A drücken und gedrückt halten, bis ein Reset ausgeführt wird.
3. Programm vom Monitor abspeichern mit S"Programmname",8,1000,5060.
4. Programm vom Monitor aus laden mit L"Programmname",8 und starten mit G5000. □



SPIELHÖLLE

Gewinnen Sie das große Geld

Laden Sie Freunde ein, um mit ihnen gemeinsam in der Spielhölle das Glück zu versuchen. Nach den Einsätzen rollt der Würfel und entscheidet über Gewinn oder Verlust.

Versuchen Sie Ihr Glück in der Spielhölle, die Ihr Commodore Plus/4 für Sie aufbaut. Auch Besitzer eines C16/116 können das Programm benutzen, es läuft auch auf diesen Rechnern.
Holen Sie Ihre Freunde zusammen und laden Sie zu einem Besuch in der Spielhölle ein. Außer dem Bankhalter können bis zu sieben Spieler teilnehmen. Je mehr, desto besser.
Nach dem Start erscheint der Titel des Spieles, dann wird nach der Anzahl der Mitspieler gefragt. Geben Sie die Zahl aller Wetter an, aber zählen Sie den Bankhalter nicht mit. Als nächstes will der Computer die Namen der Mitspielenden wissen. Hier geben Sie jeweils zwei Buchstaben ein, mehr braucht der Computer nicht zu speichern.
Jetzt wird das Spielfeld aufgebaut und der erste Spieler kann seinen Einsatz wagen. Dazu benutzt er die Cursortasten, um den blinkenden Cursor auf einem Feld seiner Wahl zu plazieren. Daraufhin gibt eine Ziffer von eins bis neun die Höhe des Einsatzes bekannt. Dieser wird gespeichert und der Spieler kann weitere Felder setzen. Es ist aber nicht möglich, daß er ein von ihm bereits gesetztes Feld nochmals belegt. Hat er genug gesetzt, gibt er das durch Eingabe der Ziffer Null bekannt. Darauf wird der nächste Spieler zur Eingabe seiner Wetten aufgefordert.
Wenn der letzte Spieler seine Einsätze getätigt hat, ist die Reihe wieder am ersten und so geht es weiter, bis keiner mehr etwas setzen will. Dies wird durch Druck auf die Taste PFEIL HOCH (↑) angezeigt.
Im unteren Feld sind in der Reihe EIN die Einsätze aller Spieler rot angezeigt. Oben rechts steht die Summe, die an die Bank zu zahlen ist. Diese Zahl wird ständig auf dem aktuellen Wert gehalten. Die restlichen Felder des unteren Tableaus enthalten noch Nullen, denn bisher hat sich ja noch nicht viel getan.
Es wird in drei Runden gespielt. An der rechten Seite erscheinen drei Würfel, die nacheinander geworfen werden. Sobald sie zur Ruhe kommen, werden die Gewinne der einzelnen Spieler errechnet und ausgezahlt. Dabei wird nach folgendem Schema vorgegangen: In der obersten Reihe wird gewonnen, wenn die drei Würfel die gleiche Augenzahl zeigen. In den ersten zwei Runden wird der Einsatz 50fach, in der dritten Runde sogar 100fach ausgezahlt. Haben Sie zum Beispiel die Fünf auf das Feld 333 gesetzt und die drei Würfel zeigen dies auch an, so werden Ihnen 250, in der dritten Runde 500 Punkte ausgezahlt. Sie können sich an Stelle der Punkte auch Pfennige, Markstücke oder sogar Hundertmarkscheine vorstellen.
Die Höhe des Gewinnes wird in der untersten Zeile angezeigt. Das Feld blinkt so lange, bis der glückliche Gewinner durch Druck auf die Leertaste die Auszahlung des Gewinns verlangt. Sofort werden die Punkte in der Reihe AUS angezeigt beziehungsweise zu den dort bereits vorhandenen Punkten dazugerechnet. Danach werden die weiteren Gewinnfelder geprüft und ausgezahlt.
In den Reihen zwei bis sieben sehen Sie immer zwei Ziffern. Diese Felder gewinnen, wenn die beiden Augenzahlen gewürfelt wurden, aber nur dann, wenn sie sich auch in der richtigen Reihenfolge befinden. Wenn die drei Würfel also die Folge 4–3–6 zeigen, gewinnen die Felder 4–3, 4–6 und 3–6, nicht aber die Felder 3–4, 6–4 und 6–3.
Zeigen die Würfel 2–3–2 an, gewinnen die Felder 2–2, 2–3 und 3–2, denn die Drei wurde sowohl vor als auch nach der Zwei gezogen. Für ein Feld mit zwei gleichen Ziffern wird der zweifache, in der dritten Runde der achtfache Einsatz ausgezahlt. Für zwei unterschiedliche Ziffern gibt es den zwei- beziehungsweise fünffachen Einsatz. Paschfelder gewinnen auch bei drei gleichen Ziffern.

VIELE GEWINNMÖGLICHKEITEN

Doch damit sind die Gewinnmöglichkeiten noch nicht erschöpft. Am rechten und unteren Rand der Gewinntafel befinden sich Felder mit Pfeilen und Schrägstrichen. Auch auf diese Felder kann gesetzt werden. Die Felder mit Pfeilen gewinnen mit jedem Feld, das in der Reihe oder Spalte gewinnt, die von den Pfeilen gezeigt werden. Das Feld mit den Schrägstrichen gewinnt, wenn ein Feld mit zwei gleichen Ziffern gewinnt. Das Pfeilfeld in der ersten Reihe erhält bei Gewinn den fünffachen Einsatz, in der dritten Runde wird 25fach ausgezahlt. Die anderen Felder gewinnen den Einsatz zurück, in der letzten Runde das Doppelte, wenn ein Pasch geworfen wurde.
Es sieht im ersten Moment so aus, als ob es sich nicht lohnen würde, auf die Randfelder zu setzen, da es im Gewinnfall ja nur den Einsatz zurückgibt. Doch Sie sollten bedenken, daß die Chance zu gewinnen größer ist. Da in drei Runden gespielt wird, kann natürlich in einer Reihe auch dreimal gewonnen werden. Es kommt noch hinzu, daß für jedes Gewinnfeld der Reihe ausgezahlt wird. Hat also ein Spieler die Reihe von 1–1 bis 1–6 gesetzt und die Würfel zeigen die Kombination 1–3–4 an, so wird in dieser Reihe zweimal gewonnen, nämlich 1–3 und 1–4. Damit werden auch die kleinen Chancen noch attraktiv.
Nachdem die Gewinne der dritten Runde ausgezahlt sind, ist das Spiel zu Ende. Die Differenz der Reihen EIN und AUS wird als Gesamtgewinn oder Verlust in die Reihe +/– übernommen. Punkte, die schwarz erscheinen, sind Gewinne, rote dagegen zeigen Verluste an und sind der Gewinn der Bank.
Geben Sie durch J oder N bekannt, ob ein weiteres Spiel laufen soll oder nicht und die Spielhölle schließt ihre Pforten oder beginnt ein neues Spiel.
Beachten Sie während des Spieles immer das Feld an der rechten Seite des Bildes. Dort erhalten Sie Hinweise, was zu tun ist oder was Sie falsch gemacht haben. So ist es sehr einfach, das Programm zu bedienen.

G. Kramer □

```
10 rem spielhoelle==========c16/p4 <io>
20 rem (p) commodore welt team      <ho>
30 rem ============================ <ng>
40 rem (c) by g. kramer             <kd>
50 rem             emsdetten        <em>
60 rem                              <ah>
70 rem basic v3.5                   <nl>
80 rem c16/116/plus4                <ki>
90 rem ============================ <jg>
100 gosub4530                       <ga>
110 gosub190                        <ko>
120 gosub1110                       <la>
130 gosub2450                       <cf>
140 ifa$="j"then120                 <hc>
150 end                             <kn>
160 ==========                      <cc>
170 bildaufbau                      <ln>
180 ==========                      <dl>
190 pokeb,85                        <kf>
200 a=64:a0=114                     <gj>
210 a1=7:b0=b+1                     <bp>
220 vol8:sound1,596,950             <hp>
230 gosub4410                       <kg>
240 pokeb+28,73                     <pi>
250 fori=1to7                       <cd>
260 fori0=0to1                      <nf>
270 a=66:ifi0=1thena=107            <km>
280 pokeb+40+(i-1)*80+40*i0,a       <dl>
290 a=32:ifi0=1thena=64             <mj>
300 a0=66:ifi0=1thena0=91           <ik>
310 b0=b+41+(i-1)*80+i0*40          <bm>
320 sound3,596,1                    <np>
330 gosub4410                       <dd>
340 a=66:ifi0=1thena=115            <ej>
350 pokeb+68+(i-1)*80+40*i0,a       <ld>
360 nexti0,i                        <on>
370 pokeb+600,66                    <bc>
380 a=32                            <dj>
390 a0=66                           <op>
400 b0=b+601                        <ni>
410 sound3,596,10                   <nm>
420 gosub4410                       <ja>
430 pokeb+628,66                    <aa>
440 pokeb+640,74                    <gk>
450 a=64:a0=113                     <ha>
460 a1=7:b0=b+641                   <aj>
470 sound2,685,200                  <on>
480 gosub4410                       <aj>
490 pokeb+668,75                    <nj>
500 fori=0to6                       <od>
510 fori0=1to6                      <bn>
520 a$=right$(str$(i0),1)           <im>
530 a0$=right$(str$(i),1)           <pb>
540 ifi=0thena$=a$+a$+a$:goto560    <nj>
550 a$=a$+"-"+a0$                   <od>
560 poke202,i0*4-3                  <hj>
570 poke205,i*2+1                   <hk>
580 sys55464                        <ld>
590 printa$                         <il>
600 nexti0,i                        <ap>
610 color1,3,6                      <mf>
620 sound2,739,250                  <ab>
630 fori=17to23                     <pd>
640 b0=b+i*40+4                     <la>
650 a=66:a0=66:a1=32:a2=66          <io>
660 if(iand1)=1thena=107:a0=115:a1
=67:a2=91                           <an>
670 ifi=17thena=112:a0=110:a1=64:a
2=114                               <hk>
680 ifi=23thena=109:a0=125:a1=64:a
2=113                               <el>
690 pokeb0,a                        <gm>
700 fori0=1tos                      <lk>
710 b1=i0*5-5                       <kb>
720 fori1=1to4                      <jb>
730 pokeb0+b1+i1,a1                 <jk>
740 next                            <ep>
750 ifi0<sthenpokeb0+b1+5,a2        <jl>
760 ifi0=sthenpokeb0+b1+5,a0        <ml>
770 nexti0,i                        <hd>
780 fori=1tos                       <hn>
790 poke202,1+5*i                   <lf>
800 poke205,17                      <ke>
810 sys55464                        <hn>
820 printn$(i)                      <eb>
830 next                            <ae>
840 sound2,810,200                  <bb>
850 print"ein"                      <jg>
860 print:print"aus"                <ip>
870 print:print"+/-"                <ol>
880 a$="^^^^"                       <dk>
890 poke205,15                      <jl>
900 fori=1to21step4                 <nl>
910 poke202,i                       <gc>
920 sys55464                        <ce>
930 printa$;                        <kl>
940 next                            <ob>
950 poke202,25                      <kh>
960 sys55464                        <ak>
970 a$=chr$(109)+chr$(109)+chr$(10
9)                                  <ho>
980 printa$                         <mh>
990 a$="___"                        <lk>
1000 fori=1to13step2                <kb>
1010 poke202,25                     <mb>
1020 poke205,i                      <oj>
1030 sys55464                       <ma>
1040 printa$                        <kn>
1050 next                           <lo>
1060 fori=8to0step-.01:voli:next    <kl>
1070 return                         <pc>
1080 ================               <ie>
1090 einsaetze machen               <lo>
1100 ================               <fi>
1110 gosub2060                      <ac>
1120 vol8                           <hp>
```

```
1130 gosub2350                              <ih>
1140 sp=1:cs=1:cz=1                         <ok>
1150 na=f+681+5*sp                          <hf>
1160 pokena,peek(na)or128                   <jn>
1170 pokena+1,peek(na+1)or128               <cn>
1180 poke205,4:poke202,30                   <oa>
1190 sys55464                               <ni>
1200 print"cursor="                         <ni>
1210 printtab(30)"auswahl"                  <dd>
1220 print                                  <ck>
1230 printtab(31)"1 - 9"                    <no>
1240 printtab(30)"einsatz"                  <eb>
1250 print                                  <gg>
1260 printtab(30)"0 ="                      <ip>
1270 printtab(30)"naechster"                <fd>
1280 printtab(32)"spieler"                  <fe>
1290 print                                  <lh>
1300 printtab(30)"^ = ende"                 <ll>
1310 fe=f+80*cz+4*cs-43                     <km>
1320 pokefe,peek(fe)or128                   <jp>
1330 pokefe+1,peek(fe+1)or128               <hf>
1340 pokefe+2,peek(fe+2)or128               <kj>
1350 poke239,0                              <mp>
1360 getkeya$:a=asc(a$)                     <bl>
1370 ifa=48then1750                         <bl>
1380 ifa=94then1950                         <bm>
1390 if(a>48)and(a<58)then1530              <cd>
1400 ifa=145thencz=cz-1:goto1450            <ca>
1410 ifa=29thencs=cs+1:goto1450             <ed>
1420 ifa=17thencz=cz+1:goto1450             <de>
1430 ifa=157thencs=cs-1:goto1450            <mb>
1440 goto1360                               <do>
1450 pokefe,peek(fe)and127                  <ia>
1460 pokefe+1,peek(fe+1)and127              <fl>
1470 pokefe+2,peek(fe+2)and127              <eh>
1480 ifcs=0thencs=7                         <ap>
1490 ifcs=8thencs=1                         <ed>
1500 ifcz=0thencz=8                         <me>
1510 ifcz=9thencz=1                         <ek>
1520 goto1310                               <ch>
1530 ifei(sp,cs,cz)=0then1650               <co>
1540 gosub2350                              <ao>
1550 poke205,5:poke202,30                   <gd>
1560 sys55464                               <mb>
1570 sound3,1023,30                         <od>
1580 printchr$(130)"das feld"               <jm>
1590 printtab(32)"ist"                      <jf>
1600 printtab(30)"bereits"                  <pc>
1610 printtab(30)"besetzt"                  <bc>
1620 getkeya$                               <df>
1630 gosub2350                              <em>
1640 goto1150                               <dl>
1650 e=val(a$)                              <ip>
1660 ei(sp,cs,cz)=e                         <hn>
1670 e(sp)=e(sp)+e                          <fg>
1680 g(0)=g(0)+e                            <hl>
1690 gosub2060                              <nl>
1700 pokefe,peek(fe)and127                  <kg>
1710 pokefe+1,peek(fe+1)and127              <en>
1720 pokefe+2,peek(fe+2)and127              <ij>
1730 cs=cs+1:ifcs=8thencs=6                 <pi>
1740 goto1310                               <ml>
1750 pokena,peek(na)and127                  <kk>
1760 pokena,peek(na)and127                  <po>
1770 pokena+1,peek(na+1)and127              <fe>
1780 pokefe,peek(fe)and127                  <jm>
1790 pokefe+1,peek(fe+1)and127              <oo>
1800 pokefe+2,peek(fe+2)and127              <cm>
1810 gosub2350                              <pe>
1820 sound1,596,60                          <li>
1830 sound2,685,60                          <ij>
1840 sp=sp+1                                <on>
1850 ifsp>sthensp=1                         <ho>
1860 poke205,5:poke202,30                   <nb>
1870 sys55464                               <hh>
1880 printchr$(130)n$(sp)" (j/n)"           <hl>
1890 poke239,0                              <eh>
1900 getkeya$                               <mg>
1910 if(a$<>"n")and(a$<>"j")then19
00                                          <km>
1920 gosub2350                              <lo>
1930 ifa$="j"then1150                       <ck>
1940 goto1820                               <ak>
1950 pokena,peek(na)and127                  <nd>
1960 pokena,peek(na)and127                  <lb>
1970 pokena+1,peek(na+1)and127              <gk>
1980 pokefe,peek(fe)and127                  <pj>
1990 pokefe+1,peek(fe+1)and127              <ke>
2000 pokefe+2,peek(fe+2)and127              <ma>
2010 gosub2350                              <pn>
2020 return                                 <gf>
2030 =================                      <fo>
2040 setzen spielwerte                      <ja>
2050 =================                      <om>
2060 forj=1tos                              <mm>
2070 sound1,800,3                           <bn>
2080 color1,3,5                             <lb>
2090 poke205,18:poke202,5*j                 <ak>
2100 sys55464                               <cm>
2110 printusing"####";e(j)                  <pb>
2120 color1,1                               <ik>
2130 poke205,20:poke202,5*j                 <bo>
2140 sys55464                               <ck>
2150 printusing"####";a(j)                  <mk>
2160 a=1:ifg(j)<0thena=3                    <np>
2170 color1,a,5                             <je>
2180 poke205,22:poke202,5*j                 <dk>
2190 sys55464                               <oh>
2200 a=abs(g(j))                            <je>
2210 printusing"####";a                     <kc>
2220 next                                   <ol>
2230 printchr$(19)                          <ni>
2240 pokeb+79,32                            <ho>
2250 color1,16,5                            <pl>
2260 printtab(30)"bank";                    <ei>
2270 printusing"#####";g(0)                 <bf>
```

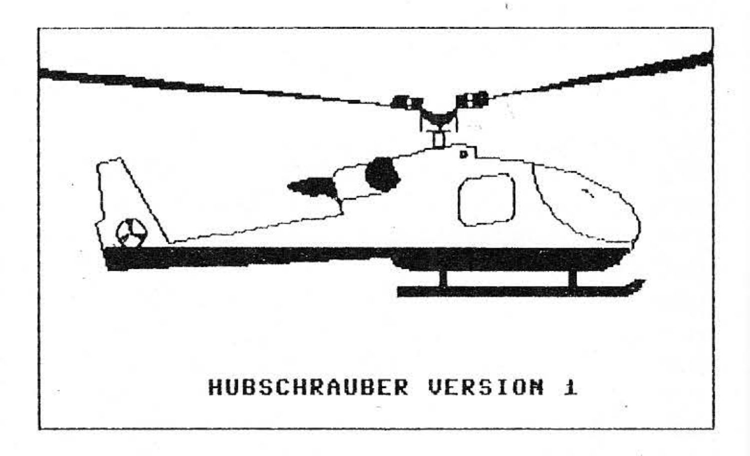
```
2280 printchr$( 19)                          <fl>
2290 printtab( 34)chr$( 148)                 <gi>
2300 color1,1                                <mj>
2310 return                                  <km>
2320 ===================                     <bo>
2330 textzeilen loeschen                     <mp>
2340 ===================                     <gl>
2350 a0$=chr$( 32)+chr$( 32)+chr$( 32
)                                            <hd>
2360 poke205,2:poke202,0:sys55464            <md>
2370 fori=2to16                              <mc>
2380 printtab( 30)a0$a0$a0$                  <hh>
2390 next                                    <ea>
2400 pokeb+119,32:pokeb+159,32               <if>
2410 pokeb+199,32:return                     <ah>
2420 ============                            <gn>
2430 spiel laeuft                            <kc>
2440 ============                            <bd>
2450 gosub2350                               <ef>
2460 fori=112to512step200                    <jp>
2470 fori0=0to160step40                      <kb>
2480 a=66:a0=32:a1=66                        <dn>
2490 ifi0=0thena=85:a0=64:a1=73              <bd>
2500 ifi0=160thena=74:a0=64:a1=75            <dh>
2510 a2=b+i+i0                               <mm>
2520 pokea2,a                                <an>
2530 fori1=1to3                              <be>
2540 pokea2+i1,a0                            <ge>
2550 next                                    <ib>
2560 pokea2+4,a1                             <ae>
2570 nexti0,i                                <ff>
2580 fori=1to3                               <bk>
2590 color1,15,5                             <jk>
2600 char1,37,3,""                           <nc>
2610 printi                                  <gl>
2620 color1,1                                <of>
2630 pokef+117,0:pokeb+117,112               <nm>
2640 pokef+118,0:pokeb+118,64                <ld>
2650 pokef+119,0:pokeb+119,110               <nk>
2660 pokef+157,0:pokeb+157,66                <im>
2670 pokef+159,0:pokeb+159,66                <na>
2680 pokef+197,0:pokeb+197,109               <aj>
2690 pokef+198,0:pokeb+198,64                <ph>
2700 pokef+199,0:pokeb+199,125               <il>
2710 fori0=3to13step5                        <nc>
2720 gosub3090                               <cg>
2730 w( i0/5+1)=a                            <nd>
2740 next                                    <pp>
2750 gosub3410                               <nk>
2760 next                                    <ch>
2770 poke205,24:poke202,4                    <ia>
2780 sys55464                                <ml>
2790 print"ende des spieles";                <kp>
2800 fori=0to8step.5                         <dh>
2810 voli                                    <ba>
2820 sound1,596,10                           <cg>
2830 sound2,685,10                           <gh>
2840 next                                    <mi>
2850 forl=960to999                           <cb>
2860 pokeb+l,32                              <mp>
2870 next                                    <ae>
2880 fori=1tos                               <ge>
2890 sound1,596,20:sound2,658,20             <ch>
2900 g( i)=g( i)-e( i)+a( i)                 <cm>
2910 e( i)=0:a( i)=0                         <go>
2920 gosub2060                               <bn>
2930 fori0=0to7                              <lh>
2940 fori1=0to8                              <ch>
2950 ei( i,i0,i1)=0                          <kb>
2960 nexti1,i0,i                             <me>
2970 gosub2350                               <ie>
2980 poke205,5:poke202,30:sys55464           <cb>
2990 print"ein neues"                        <ig>
3000 printtab( 32)"spiel?"                   <fh>
3010 print                                   <de>
3020 printchr$( 130)tab( 32)"( j/n)"         <nk>
3030 getkeya$                                <kn>
3040 if( a$<>"j")and( a$<>"n")then30
30                                           <bo>
3050 return                                  <hj>
3060 ========                                <ce>
3070 wuerfeln                                <ln>
3080 ========                                <ki>
3090 a$=chr$( 32)                            <ob>
3100 a0$=chr$( 119)                          <bg>
3110 w$=a$+a$+a$                             <mk>
3120 w0$=a$+a0$+a$                           <an>
3130 w1$=a$+a$+a0$                           <go>
3140 w2$=a0$+a$+a$                           <bb>
3150 w3$=a0$+a$+a0$                          <ck>
3160 forl=1to50                              <kf>
3170 a=1+int( rnd( a)*6)                     <nl>
3180 a$=w$                                   <nb>
3190 ifa>1thena$=w2$                         <ej>
3200 ifa>3thena$=w3$                         <oj>
3210 poke205,i0:poke202,33                   <fn>
3220 sys55464                                <ga>
3230 sound3,600,2:sound1,600,2               <fd>
3240 printa$                                 <ko>
3250 ifa=6then3280                           <ef>
3260 a$=w$                                   <oh>
3270 if( aand1)=1thena$=w0$                  <hh>
3280 printtab( 33)a$                         <aa>
3290 ifa=6then3350                           <ip>
3300 a$=w$                                   <nm>
3310 ifa=1then3350                           <ck>
3320 a$=w1$                                  <nm>
3330 ifa<4then3350                           <ci>
3340 a$=w3$                                  <bn>
3350 printtab( 33)a$                         <fk>
3360 next                                    <nm>
3370 return                                  <pm>
3380 ==============                          <jn>
3390 pruefen gewinn                          <el>
3400 ==============                          <ki>
3410 a4=0                                    <ga>
```

```
3420 ifw(1)<>w(2)then3500            <be>
3430 ifw(1)<>w(3)then3500            <kd>
3440 a0=w(1):a1=0:a2=50              <df>
3450 ifi=3thena2=100                 <fo>
3460 gosub3900                       <bj>
3470 a0=7:a1=0:a2=5                  <dp>
3480 ifi=3thena2=25                  <hf>
3490 gosub3900                       <nf>
3500 a0=0                            <pl>
3510 ifw(1)=w(2)thena0=w(1)          <bl>
3520 ifw(1)=w(3)thena0=w(1)          <jk>
3530 ifw(2)=w(3)thena0=w(2)          <in>
3540 a2=3:ifi=3thena2=8              <nj>
3550 ifa0=0then3590                  <ag>
3560 a1=a0                           <mj>
3570 gosub3900                       <nh>
3580 gosub4190                       <po>
3590 a0=w(1):a1=w(2)                 <mf>
3600 a2=2:ifi=3thena2=6              <gg>
3610 ifa0<>a1thengosub3900:gosub41
90                                   <fk>
3620 a1=w(3)                         <jf>
3630 ifa1=w(2)then3660               <pe>
3640 a2=2:ifi=3thena2=6              <ia>
3650 ifa0<>a1thengosub3900:gosub41
90                                   <bk>
3660 a0=w(2)                         <gn>
3670 ifa0=w(1)then3700               <kb>
3680 a2=2:ifi=3thena2=6              <lc>
3690 ifa0<>a1thengosub3900:gosub41
90                                   <kp>
3700 ifa4>0then3790                  <da>
3710 poke205,24:poke202,4            <ge>
3720 sys55464                        <oh>
3730 print"leider kein gewinn";      <ig>
3740 poke239,0                       <lh>
3750 getkeya$                        <nk>
3760 forl=960to999                   <ol>
3770 pokeb+l,32                      <ke>
3780 next                            <ch>
3790 a$=chr$(32)+chr$(32)+chr$(32)   <po>
3800 forl=3to13step5                 <fo>
3810 poke205,l:poke202,0:sys55464    <fa>
3820 printtab(33)a$                  <co>
3830 printtab(33)a$                  <oe>
3840 printtab(33)a$                  <fi>
3850 next                            <le>
3860 return                          <ne>
3870 =========                       <ja>
3880 auszahlen                       <pf>
3890 =========                       <hi>
3900 f0=f+37+4*a0+80*a1              <hg>
3910 forl=0to2                       <ph>
3920 pokef0+l,peek(f0)or128          <mo>
3930 next                            <fe>
3940 forl=1tos                       <jg>
3950 a3=ei(l,a0,a1+1)                <ke>
3960 ifa3=0then4110                  <jf>
3970 poke205,24:poke202,4            <mi>
3980 sys55464                        <ce>
3990 printn$(1)" gewinnt"a2"mal"a3
;                                    <le>
4000 a4=a2                           <cg>
4010 poke239,0                       <gl>
4020 getkeya$                        <ml>
4030 forl0=1toa3                     <ap>
4040 g(0)=g(0)-1*a2                  <ln>
4050 a(l)=a(l)+1*a2                  <ho>
4060 gosub2060                       <db>
4070 next                            <gn>
4080 forl0=b+960tob+999              <gl>
4090 pokel0,32                       <ma>
4100 next                            <kk>
4110 next                            <lo>
4120 forl0=0to2                      <ma>
4130 pokef0+l0,peek(f0)and127        <ci>
4140 next                            <pk>
4150 return                          <bk>
4160 ===================             <op>
4170 sammelfelder pruefen            <mm>
4180 ===================             <cf>
4190 ifi<3then4340                   <eo>
4200 f1=f0                           <hc>
4210 forl=0to2                       <fn>
4220 pokef1+l,peek(f1)or128          <kf>
4230 next                            <kp>
4240 b0=a0:b1=a1                     <dm>
4250 a2=2                            <bc>
4260 ifa0<>a1then4290                <po>
4270 a0=7:a1=7                       <de>
4280 gosub3900                       <om>
4290 a0=b0:a1=7                      <gk>
4300 gosub3900                       <hd>
4310 a0=7:a1=b1                      <fe>
4320 gosub3900                       <on>
4330 a0=b0:a1=b1                     <oj>
4340 forl0=0to2                      <lg>
4350 pokef1+l0,peek(f1)and127        <am>
4360 next                            <le>
4370 return                          <ne>
4380 =============                   <hk>
4390 linien ziehen                   <ha>
4400 =============                   <pf>
4410 forl=1toa1                      <dg>
4420 forl0=1to3                      <cd>
4430 pokeb0+(l-1)*4+l0-1,a           <mb>
4440 next                            <fe>
4450 pokeb0+l*4-1,a0                 <lg>
4460 next                            <hm>
4470 forl=0to2                       <jo>
4480 pokeb0+4*a1-1,a                 <kb>
4490 return                          <mf>
4500 ============                    <nb>
4510 anfangswerte                    <pe>
4520 ============                    <jd>
4530 color1,5,3    :rem  zeichenfar
```

```
be                                        <pk>
4540 scnclr         :rem  bildschirm <ph>
4550 color0,14,7    :rem  hintergrun
d                                         <ib>
4560 color4,14,5    :rem  rand        <jh>
4570 b=3072         :rem  bildadress
e                                         <ab>
4580 f=2048         :rem  farbram     <ej>
4590 s=7            :rem  spielerzah
l                                         <nf>
4600 dimn$(s)       :rem  spielernam
en                                        <fg>
4610 dime(s)        :rem  einsatz     <ga>
4620 dima(s)        :rem  gewinn      <kl>
4630 dimg(s)        :rem  gesamt      <il>
4640 dimei(s,7,8)   :rem  felder ges
etzt                                      <jj>
4650 dimw(3)        :rem  wuerfe      <ai>
4660 a$=chr$(32)    :rem  wuerfel     <kn>
4670 a0$=chr$(119)                        <hf>
4680 w$=a$+a$+a$                          <ih>
4690 w0$=a$+a0$+a$                        <lm>
4700 w1$=a$+a$+a0$                        <hh>
4710 w2$=a0$+a$+a$                        <be>
4720 w3$=a0$+a$+a0$                       <ao>
4730 fori=40to280step240                  <ga>
4740 b0=b+i                               <bl>
4750 vol8                                 <ha>
4760 fori0=0to160step40                   <op>
4770 fori1=1to3                           <kd>
4780 reada$                               <lg>
4790 a=dec(a$)                            <kh>
4800 fori2=7to0step-1                     <ni>
4810 if(aand2^i2)=0then4840               <nk>
4820 sound2,950,5                         <gp>
4830 pokeb0+i0+8*i1+7-i2,102              <de>
4840 nexti2,i1,i0,i                       <ek>
4850 vol0                                 <ph>
4860 poke202,10                           <ag>
4870 poke205,16                           <ko>
4880 sys55464                             <ac>
4890 print"ein wuerfelspiel von"          <ik>
4900 a$="g  k r a m e r"                  <go>
4910 poke202,13                           <fn>
4920 poke205,19                           <ik>
4930 sys55464                             <hh>
4940 fori=1to14                           <mc>
4950 ifmid$(a$,i,1)=chr$(32)then49
90                                        <ic>
4960 vol8                                 <ll>
4970 sound1,500,5                         <ff>
4980 fori0=8to0step-1:voli0:next          <ch>
4990 printmid$(a$,i,1);                   <bb>
5000 next                                 <lj>
5010 getkeya$                             <pb>
5020 vol0                                 <ke>
5030 scnclr                               <fl>
5040 print                                <ca>
5050 print"wieviele spieler? (1-7)
"                                         <ea>
5060 getkeya$                             <am>
5070 if(a$<"1")or(a$>"7")then5060         <me>
5080 s=val(a$)                            <ne>
5090 print                                <ie>
5100 fori=1tos                            <jb>
5110 print"name spieler"i;                <pm>
5120 a0$=""                               <go>
5130 getkeya$                             <hj>
5140 if(a$<"a")or(a$>"z")then5130         <kc>
5150 a0$=a0$+a$                           <dd>
5160 printa$;                             <ba>
5170 iflen(a0$)<2then5130                 <ni>
5180 n$(i)=a0$                            <fj>
5190 print                                <en>
5200 next                                 <ek>
5210 return                               <gk>
5220 data77,2f,40,84,a8,40                <de>
5230 data67,2e,47,14,28,40                <im>
5240 datae4,2f,78                         <dm>
5250 data94,a1,0f,93,21,08                <ha>
5260 dataf4,a1,0e,94,a1,08                <im>
5270 data93,3d,ef                         <fm>
5280 =p=r=o=g=r=a=m=m=e=n=d=e=====        <mm>
```

HUBSCHRAUBER C16

Grafik-Demo



Selbst mit dem begrenzten Speicher eines C16 läßt sich das Abbild eines Hubschraubers auf den Bildschirm bringen. Mit Drucker und Hardcopyroutinen dürfen Sie ihn auch schwarz auf weiß auf Papier bewundern.

Hubschrauber C16 ist ein Grafik-Programm, das einen Hubschrauber auf dem Bildschirm darstellt. Das Be-

sondere daran ist, daß eine Grafik dieser Klasse auch für einen C16/116 mit 16 KByte bestimmt ist, obwohl der Speicherplatz im Grafikmodus doch sehr bescheiden ist. An Peripherie wird neben dem C16/116 oder dem Plus/4 lediglich ein Speicherwerk zum Sichern des Programms benötigt.
Nach dem Starten werden Sie in zirka zwölf Sekunden die Hubschraubergrafik auf dem Bildschirm bewundern können. Sind Sie im Besitz eines Druckers mit einer entsprechenden Hardcopyroutine (in der C16/P/4 SPECIAL Nr. 1 wurde eine Hardcopyroutine für Commodore-Drucker vorgestellt), so können Sie diese wunderschöne Grafik auch zu Papier bringen. □

```
10 rem hubschrauber============c16  <dm>
20 rem (p) commodore welt team      <ho>
30 rem (c) by ralf haack            <fd>
40 rem c16/116/plus4                <kc>
50 rem ========================     <db>
60 color0,2,7:color1,1,1:color4,15
,6                                  <po>
70 graphic1,1:draw1,0,20to166,38    <gf>
80 draw1,0,25to166,38:paint1,0,23:
draw1,0,23to166,38:draw1,320,11to2
13,35                               <cf>
90 draw1,320,18to213,35             <lf>
100 paint1,319,15:draw1,320,15to21
3,35:box1,167,35,181,41,8,1         <nm>
110 box1,167,35,181,41,08,1         <mg>
120 draw0,168,34:draw0,167,40       <pa>
130 draw0,175,37to175,36            <ma>
140 draw0,176,36to176,37            <jg>
150 draw0,176,39to176,40            <nj>
160 draw0,175,40to175,39            <hn>
170 box1,198,33,212,40,170,1:draw0
,213,32                             <bp>
180 draw0,214,38to214,37            <pp>
190 draw0,205,34to205,35            <nn>
200 draw0,205,38to205,37            <mj>
210 draw0,204,34to204,35            <pn>
220 draw0,204,37to204,38            <hg>
230 draw1,28,68to39,65              <bc>
240 draw1,28,68to31,96:drawto27,98
:drawto33,122                       <cp>
250 draw1,39,65to62,108:drawto144,
93                                  <ha>
260 draw1,33,122to166,115:circle1,
44,103,7                            <gf>
270 draw1,44,103to44,96:draw1,44,1
03to42,97:paint1,43,98              <ed>
280 draw1,44,103to38,106            <em>
290 draw1,44,103to40,108            <fi>
300 draw1,44,103to39,107            <kp>
310 draw1,44,103to48,106            <me>
320 draw1,44,103to51,104:paint1,50
,105                                <cb>
330 draw1,44,103to50,105            <ao>
340 draw1,144,93to142,86            <mf>
350 drawto135,83:drawto130,84:draw
to119,81:drawto118,78:drawto129,75
:drawto139,75                       <lh>
360 drawto142,86:paint1,130,80      <el>
370 draw1,139,75to142,70:drawto160
,67                                 <nn>
380 circle1,163,73,8,,110,382       <bc>
390 drawto171,76:paint1,163,73      <nd>
400 draw1,142,86to165,82            <fm>
410 draw1,166,66to184,60            <dg>
420 drawto199,58:drawto207,59:draw
to207,65                            <dd>
430 circle1,207,100,77,35,0,130     <jo>
440 draw1,166,115to170,120          <ga>
450 drawto178,122:drawto266,122     <lo>
460 draw1,32,113to279,113:paint1,3
4,114                               <od>
470 draw1,32,111to279,111           <kh>
480 draw1,32,110to279,110           <jk>
490 box1,190,122,193,130,0,1        <hi>
500 box1,251,122,254,130,0,1        <ol>
510 box1,170,130,274,134,0,1        <ak>
520 circle1,274,126,8,4,90,180      <hi>
530 circle1,274,126,12,8,90,180     <hi>
540 draw1,282,126to287,126:paint1,
276,132                             <lk>
550 circle1,274,67,40,,168,270      <gb>
560 circle1,254,90,10,,20,50        <lj>
570 circle1,205,80,5,,260,360       <ee>
580 draw1,205,75to222,73            <bn>
590 circle1,220,78,5,,,100          <ho>
600 draw1,225,79to225,92            <mb>
610 circle1,220,92,5,,90,190        <hd>
620 drawto204,99                    <ba>
630 circle1,204,94,5,,180,280       <ad>
640 drawto199,80                    <ak>
650 draw1,181,41to181,51            <gi>
660 draw1,198,39to198,51            <ge>
670 draw1,184,51to195,51:box1,187,
52,192,60                           <kh>
680 draw1,190,50to182,44            <ie>
690 draw1,190,50to197,44            <oa>
700 circle1,190,27,16,,150,210:pai
nt1,190,45                          <mg>
710 circle1,201,63,2:draw1,201,63   <ce>
720 draw0,199,61to199,65            <hh>
730 draw1,200,63:box1,0,0,319,199   <fc>
740 char1,10,22,"hubschrauber vers
ion 1"                              <oc>
750 rem =p=r=o=g=r=a=m=m=e=n=d=e==  <ak>
```

BUDGET-ANALYSE

Finanz-minister

Gezielte Einsparungen sind der Nutzen dieses Programmes, mit dem Sie sich den Überblick über Ihre Ein- und Ausgaben verschaffen können.

Ein gravierender Nachteil des C16 gegenüber seinem Schwestermodell Plus/4 ist sicher der Verzicht auf die eingebaute Software.
Für Privatpersonen dürfte vor allem die Datenbank interessant sein, weil sie eine kontrollierte Haushaltsführung ermöglicht.
Das Programm „Budget-Analyse" geht noch einen Schritt weiter, indem es einen Datenvergleich zuläßt, der im Rahmen einer normalen Datenbank nicht möglich ist. Für den Gebrauch ist lediglich der C16 in seiner Grundversion und ein Diskettenlaufwerk nötig. Natürlich ist das Programm ebenso auf dem Plus/4 und dem C16 mit Speichererweiterung lauffähig.

BEDIENUNGSANLEITUNG

Budget-Analyse soll Ihnen dazu verhelfen, Klarheit und Ordnung in Ihre persönlichen Finanzen zu bringen. Hierzu werden alle Ein- und Ausgaben in Unterposten aufgeteilt und registriert, um sie zusammen mit den Ein- und Ausgabesummen sowie dem Kontostand abzuspeichern und abrufbar zu machen.
Der Datenabruf kann in verschiedenen Formen erfolgen, wobei sich die Daten zur Finanzanalyse gegenüberstellen lassen.

- Tippen Sie das Listing wie gewohnt ab und speichern Sie das Programm auf einer formatierten Diskette.
- Lassen Sie das Diskettenlaufwerk zum Programmbetrieb an.
- Nach Start mit RUN erscheint das Hauptmenü.
- Anfangs ist es nötig, das Programm für Ihre persönlichen Bedürfnisse einzurichten. Wählen Sie hierzu den Menüpunkt „#" **(Einrichtung)**.

Danach wird Ihnen der maßgebliche Programmabschnitt aufgelistet.
Von den insgesamt 19 abrufbaren Einzelposten lassen sich 15 von Ihnen frei definieren. Die restlichen vier Posten sind für Einnahmesumme, Ausgabensumme, Bilanz und Kontostand reserviert.
Die Einrichtung erfolgt durch Ausfüllen der gelisteten DATA-Zeilen:
In Zeile 1840 ist die Anzahl der vorgesehenen Einnahmen und Ausgaben anzugeben (insgesamt maximal 15).
Weil die Dateneingabe später mit Hilfe von Kennbuchstaben erfolgt, müssen Sie innerhalb der Zeilen 1860 bis 2000 zunächst das X durch einen Kennbuchstaben ersetzen und dahinter die Bezeichnung des betreffenden Einzelpostens eintragen. Es ist zweckmäßig, wenn sich der Kennbuchstabe mit dem Posten in Verbindung bringen läßt, zum Beispiel K für Kleidung.

Nicht benötigte DATA-Zeilen können offen gelassen werden.
Nach Abschluß der Einrichtung können Sie das Programm erneut mit RUN starten.
Für die Menüpunkte 1 bis 5 erscheinen nach Abruf in einer grünen Zeile am oberen Bildrand **Untermenüpunkte:**
* führt zurück zum Hauptmenü,
↑ bewirkt den Ausdruck der betreffenden Bildschirmseite,
löscht die Daten der angezeigten Bildschirmseite.

VARIABLENLISTE

Dimensionierte Variablen:

K% (12,20)	= File-Daten		
	K% (M,16)	=	Kontostand
	K% (M,17)	=	Summe Einnahmen
	K% (M,18)	=	Summe Ausgaben
	K% (M,19)	=	Bilanz
	K% (M,20)	=	Anzahl Notizen

Dimensionierte Strings:

C$ (19)	= Postenbezeichnungen
M$ (12)	= Monatsbezeichnungen
N$ (12,11)	= Notizen
KC$ (19)	= Kennbuchstaben

Undimensionierte Variablen:

A,B,C	= Universelle FOR-NEXT-Variablen
AA	= Anzahl Ausgaben
AE	= Anzahl Einnahmen
HP	= Horizontale Position
KI	= INPUT für K%()
M	= Monat
NV	= Markierung für Überlauf/ Notizspeicher
TP	= Titelposition

Undimensionierte Strings:

E$	= Eingabe Notizen
J$	= Eingabe Jahr
JV$	= Reserve für J$
S$	= SAVE-String
T$	= Tastaturabfrage
X$	= Abgriff aus N$
Z$	= Leerstring

Weitere Untermenüpunkte werden im jeweiligen Zusammenhang erklärt.
Das **Laden und Abspeichern von Daten** erfolgt völlig automatisch in Form kleiner sequentieller Dateien.
Zur **Eingabe Ihrer Daten** wählen Sie Punkt 1 im Hauptmenü.
Nach Angabe des gewünschten Monats und Jahres erscheinen alle Einzelposten-Bezeichnungen, farblich nach Einnahmen und Ausgaben geordnet, zusammen mit ihren Kennbuchstaben.
Nach Wahl eines Kennbuchstabens erscheint ein INPUT an betreffender Stelle. Sie können dahinter

Bitte lesen Sie weiter auf Seite 60

```
10 rem budget-analyse==========c16 <le>
20 rem (p) commodore welt team     <ho>
30 rem ========================== <ng>
40 rem (c) by peter bergen         <pk>
50 rem                             <pd>
60 rem                             <ah>
70 rem basic v3.5                  <nl>
80 rem c16/116/plus4               <ki>
90 rem ========================== <jg>
100 gosub2350:dimn$(12,11),m$(12),
kc$(19),c$(19),k%(12,20)           <nc>
110 z$=b$+b$+b$+b4$+b4$:vol5       <bj>
120 gosub2180                      <mg>
130 gosub1740                      <ch>
140 poke239,0:getkeyt$             <ca>
150 ift$="#"thenscnclr:list1830-20
00:printchr$(145)chr$(145):stop    <hb>
160 ift$="1"then230                <ee>
170 ift$="2"then990                <nm>
180 ift$="3"then1190               <pm>
190 ift$="4"then1090               <ef>
200 ift$="5"then1440               <ol>
210 goto140                        <mo>
220 rem kosteneingabe              <ha>
230 scnclr:color1,9,6:printchr$(18
)" kosteneingabe"b$b$b5$           <ed>
240 hp=20:gosub750:gosub800        <da>
250 color1,6,6:char1,0,1,chr$(18):
print" *=menue"b2$"@=abschluss"b2$
"#=clear"b2$"&=notiz               <bm>
260 gosub2220                      <gg>
270 poke239,0:getkeyt$             <pe>
280 b=1:fora=1toae:ifkc$(b)=t$then
gosub430                           <pf>
290 b=b+1:next                     <kf>
300 fora=1toaa:ifkc$(b)=t$thengosu
b460                               <hc>
310 b=b+1:next                     <il>
320 ift$="*"then130                <ob>
330 ift$="#"thenfora=1to20:k%(m,a)
=0:next:sound1,500,5               <lh>
340 ift$="@"thengosub510:goto380   <fh>
350 ift$="&"thengosub830:goto250   <cd>
360 ifkc$(16)=t$thencolor1,2:char1
,20,23,"":printrn$;:inputk%(m,16):
kp=23:gosub490                     <nh>
370 goto270                        <no>
380 color1,6,6:char1,0,1,chr$(18):
print"*=menue"b2$d$"=copy"b$b$b4$  <pa>
390 poke239,0:getkeyt$             <co>
400 ift$="*"thengosub1550:goto130  <ia>
410 ift$=d$thengosub1650           <lc>
420 goto390                        <bf>
430 ifb/2=int(b/2)thencolor1,7,4:e
lsecolor1,7,6                      <lh>
440 char1,20,b+3,rn$:inputki:k%(m,
b)=k%(m,b)+ki:k%(m,17)=k%(m,17)+ki <ig>
450 kp=b+3:gosub490:return         <pm>
460 ifb/2=int(b/2)thencolor1,3,4:e
lsecolor1,3,6                      <gg>
470 char1,20,b+5,rn$:inputki:k%(m,
b)=k%(m,b)+ki:k%(m,18)=k%(m,18)+ki <jb>
480 kp=b+5:gosub490:return         <lp>
490 char1,20,kp,b$+b4$+b3$:return  <ng>
500 rem abrechnung                 <kn>
510 b=1:fora=1toae                 <pj>
520 ifb/2=int(b/2)thencolor1,7,4:e
lsecolor1,7,6                      <fc>
530 char1,hp,b+3,chr$(18):printusi
ng"######";k%(m,b)                 <nc>
540 b=b+1:next:color1,7,3:char1,hp
,b+3,chr$(18):printusing"######";k
%(m,17)                            <ga>
550 char1,hp+6,b+3,chr$(18):printb
3$                                 <ih>
560 fora=1toaa                     <ii>
570 ifb/2=int(b/2)thencolor1,3,4:e
lsecolor1,3,6                      <cj>
580 char1,hp,b+5,chr$(18):printusi
ng"######";k%(m,b)                 <dk>
590 b=b+1:next:color1,3,3:char1,hp
,b+5,chr$(18):printusing"######";k
%(m,18)                            <aj>
600 char1,hp+6,b+5,chr$(18)+b3$:go
sub610:gosub710:return             <gj>
610 color1,7,3:char1,hp+8,3,"%":b=
1:fora=1toae                       <ph>
620 ifb/2=int(b/2)thencolor1,7,4:e
lsecolor1,7,6                      <mo>
630 ifk%(m,b)=0thenchar1,hp+6,b+3,
chr$(18):print" 00":goto650        <ce>
640 char1,hp+7,b+3,chr$(18):printu
sing"##";int(((k%(m,b)/k%(m,17))*1
00)+.5)                            <mh>
650 char1,hp+6,b+3,chr$(18):print"
 ":b=b+1:next                      <jp>
660 color1,3,3:char1,hp+8,b+4,"%":
fora=1toaa                         <em>
670 ifb/2=int(b/2)thencolor1,3,4:e
lsecolor1,3,6                      <gm>
680 ifk%(m,b)=0thenchar1,hp+6,b+5,
chr$(18):print" 00":goto700        <hn>
690 char1,hp+7,b+5,chr$(18):printu
sing"##";int(((k%(m,b)/k%(m,18))*1
00)+.5)                            <hd>
700 char1,hp+6,b+5,chr$(18):print"
 ":b=b+1:next:return               <cb>
710 color1,2:char1,hp,23,chr$(18):
printusing"######";k%(m,16)        <ai>
720 k%(m,19)=k%(m,17)-k%(m,18)     <ge>
730 color1,8,6:char1,hp,22,chr$(18
):printusing"######";k%(m,19)      <bb>
740 return                         <fl>
750 color1,2:char1,0,2,"":input"mo
nat";m:ifm>12orm<1thensound1,100,1
0:goto750                          <oj>
```

```
760 char1,0,2,z$:char1,0,2,"":inpu
t"jahr ";j$:iflen(j$)>2thensound1,
100,10:goto760                      <na>
770 ifj$=jv$thenreturn:elsejv$=j$:
gosub1610                           <no>
780 ifds<>0thenfora=1to12:forb=1to
20:k%(a,b)=0:next:next              <bh>
790 return                          <ma>
800 char1,0,2,z$:color1,9,6:char1,
hp+1,0,chr$(18):printusing"##";m    <ad>
810 char1,hp+5,0,chr$(18):printusi
ng"####";j$:return                  <gh>
820 rem notizeneingabe              <ce>
830 color1,6,6:char1,0,1,chr$(18):
print"*=return"b2$"#=clear"b2$"not
izeneingabe "b5$                    <di>
840 color1,2:nv=0:poke2022,2:scncl
r:ifk%(m,20)=0thenk%(m,20)=1:goto8
60                                  <lf>
850 fora=1tok%(m,20):char1,2,a+4,n
$(m,a):next                         <ig>
860 ifk%(m,20)>10thengosub930       <kc>
870 char1,0,k%(m,20)+4,"":input"";
e$                                  <da>
880 ife$="#"thenfora=1to10:n$(m,a)
="":k%(m,20)=0:sound1,500,5:goto84
0                                   <fh>
890 ife$="*"thenn$(m,a)=" ":poke20
22,2:scnclr:poke2022,0:return       <ba>
900 ifnv=1ande$<>"*"then870         <hh>
910 iflen(e$)>37thengosub940:goto8
70                                  <be>
920 n$(m,k%(m,20))=e$:k%(m,20)=k%(
m,20)+1:goto860                     <mg>
930 char1,8,2,"":printchr$(18)chr$
(130)" notizspeicher voll! ":nv=1:
return                              <ld>
940 char1,10,2,"":printchr$(18)chr
$(130)" notiz zu lang! "            <el>
950 sound1,100,10:fora=1to500:next  <lp>
960 char1,10,2,b$+b4$+b3$:poke2022
,k%(m,20)+4:scnclr:return           <on>
970 return                          <cj>
980 rem darstellung monatsbezogen   <pm>
990 scnclr:color1,9,6:printchr$(18
)" monatsuebersicht"b$b$b2$         <gn>
1000 color1,6,6:printchr$(18)"*=me
nue"b2$d$"=copy"b2$"1,2=monat"b$b3
$                                   <ho>
1010 gosub2220                      <gm>
1020 poke239,0:getkeyt$             <go>
1030 ift$="*"then130                <ke>
1040 ift$=d$thengosub1650           <ck>
1050 ift$="1"thenhp=20:gosub750:go
sub800:gosub510                     <fc>
1060 ift$="2"thenhp=30:gosub750:go
sub800:gosub510                     <li>
1070 goto1020                       <lf>
```

```
1080 rem notizendisplay                 <ec>
1090 scnclr:color1,9,6:char1,0,0,c
hr$(18):print" notizendisplay"b$b$
b4$                                     <ep>
1100 hp=20:gosub750:gosub800            <mb>
1110 color1,6,6:char1,0,1,chr$(18)
:print"*=menue"b2$d$"=copy"b$b$b4$      <kc>
1120 fora=1tok%(m,20):ifa/2=int(a/
2)thencolor1,2,4:elsecolor1,2,6         <fl>
1130 char1,1,a+4,chr$(18):printz$:
char1,1,a+4,chr$(18):printn$(m,a):
next                                    <bo>
1140 poke239,0:getkeyt$                 <jk>
1150 ift$="*"then130                    <oh>
1160 ift$=d$thengosub1650               <lf>
1170 goto1140                           <fa>
1180 rem postendisplay                  <kc>
1190 scnclr:color1,9,6:char1,0,0,c
hr$(18):print" postendisplay"b$b$b
5$                                      <ea>
1200 hp=22:tp=0:gosub1270               <il>
1210 poke239,0:getkeyt$                 <he>
1220 ift$="*"then130                    <he>
1230 ift$=d$thengosub1650               <il>
1240 ift$="1"thenhp=22:tp=0:gosub1
270                                     <cl>
1250 ift$="2"thenhp=29:tp=20:gosub
1270                                    <ng>
1260 goto1210                           <id>
1270 color1,6,6:char1,0,1,chr$(18)
:print"*=menue"b2$"artikelwahl mit
 kennbuchstaben"                        <jd>
1280 gosub2220                          <gg>
1290 poke239,0:getkeyt$                 <pe>
1300 ift$="*"then130                    <bm>
1310 kv=0:fora=1to19:ift$=kc$(a)th
enkv=a                                  <cp>
1320 next:ifkv>0then1330:else1290       <ho>
1330 color1,9,6:char1,tp,0,chr$(18
):printc$(kv):gosub1400                 <pi>
1340 color1,2:gosub760:char1,0,2,b
$+b2$                                   <oj>
1350 color1,9,6:char1,tp+16,0,chr$
(18):printj$                            <bp>
1360 color1,6,6:char1,0,1,chr$(18)
:print"*=menue"b2$d$"=copy"b2$"1,2
=artikel "b$                            <hm>
1370 gosub1380:return                   <lb>
1380 fora=1to12:ifa/2=int(a/2)then
color1,4,4:elsecolor1,4,6               <gi>
1390 char1,hp,a+3,chr$(18):printus
ing"######";k%(a,kv):next:return        <of>
1400 poke2022,3:poke2024,20:scnclr
:printchr$(19)chr$(19)                  <ja>
1410 fora=1to12:ifa/2=int(a/2)then
color1,4,4:elsecolor1,4,6               <ol>
1420 char1,1,a+3,chr$(18):printb2$
m$(a)b$:next:return                     <nc>
```

```
1430 rem jahresuebersicht                <mj>
1440 scnclr:color1,9,6:char1,0,0,c
hr$(18):print" jahresuebersicht"b$
b$b2$                                    <ja>
1450 gosub760:char1,18,0,chr$(18):
printj$                                  <cn>
1460 color1,6,6:char1,0,1,chr$(18)
:print"*=menue"b2$d$"=copy"b$b$b4$ <jf>
1470 gosub2220:char1,0,2,z$:char1,
0,3,z$:char1,0,ae+5,z$:char1,0,22,
z$:char1,0,23,z$                         <bi>
1480 hp=-3:form=1to12:gosub610:hp=
hp+3:next                                <ji>
1490 color1,2:char1,4,23,"ja fe ma
 ap ma ju ju au se ok no de"             <lk>
1500 poke239,0:getkeyt$                  <po>
1510 ift$="*"then130                     <kn>
1520 ift$=d$thengosub1650                <gb>
1530 goto1500                            <hh>
1540 rem disc-save                       <ih>
1550 s$="s:"+j$+"-file":open15,8,1
5,s$:close15                             <jh>
1560 forb=1to12:forc=1to10:x$=left
$(n$(b,c),1):ifasc(x$)=0thenn$(b,c
)="*"                                    <mk>
1570 next:next:s$=j$+"-file,s,w":o
pen 2,8,2,s$                             <hd>
1580 forb=1to12:forc=1to20:print#2
,k%(b,c):next:next                       <mj>
1590 forb=1to12:forc=1to10:print#2
,n$(b,c):next:next:close2:return    <bp>
1600 rem disc-load                       <am>
1610 s$=j$+"-file,s,r":open1,8,0,s
$                                        <pc>
1620 forb=1to12:forc=1to20:input#1
,k%(b,c):next:next                       <hj>
1630 forb=1to12:forc=1to10:input#1
,n$(b,c):next:next:close1:return    <fm>
1640 rem copy                            <pp>
1650 color1,6,6:char1,11,1,"":prin
tchr$(18)chr$(130)"copy"                 <ic>
1660 open4,4:open3,3:char1,0,0,""   <ej>
1670 forz=0to24:c$="":k=40:fors=1t
ok:get#3,a$:c$=c$+a$                     <ff>
1680 ifasc(a$)=13thens=s-1:c$=""    <gd>
1690 nexts                               <nk>
1700 ifmid$(c$,k,1)=" "then:k=k-1:
c$=left$(c$,k):ifk>0then1700             <hl>
1710 print#4,c$:nextz:close3:close
4                                        <pa>
1720 color1,6,6:char1,11,1,"":prin
tchr$(18)"copy":return                   <ad>
1730 rem hauptmenue                      <hb>
1740 color0,1:color4,1:scnclr            <ie>
1750 color1,9,6:printrn$"$$$$$$$$
$$$ budget-analyse $$$$$$$$$$$$":c
olor1,2                                  <ea>
1760 char1,5,5,"# = einrichtung"    <jf>
1770 char1,5,7,"1 = eingabe kosten
/notizen"                                <oj>
1780 char1,5,11,"2 = monatsdisplay
"                                        <il>
1790 char1,5,13,"3 = postendisplay
"                                        <hk>
1800 char1,5,15,"4 = notizendispla
y"                                       <nl>
1810 char1,5,17,"5 = jahresuebersi
cht"                                     <af>
1820 return                              <ne>
1830 rem anzahl einnahmen, ausgabe
n                                        <ea>
1840 data6,9                             <pf>
1850 rem kb,bezeichnung (max.15 bu
chst.)                                   <mk>
1860 data"p","gehalt peter    "     <fp>
1870 data"b","gehalt beate    "     <lf>
1880 data"f","foto            "     <mm>
1890 data"c","computer        "     <jm>
1900 data"g","gagen           "     <cc>
1910 data"z","zuschuesse      "     <ea>
1920 data"x","feste kosten    "     <ko>
1930 data"t","telefon         "     <nh>
1940 data"l","lebensmittel    "     <na>
1950 data"k","kleidung        "     <og>
1960 data"a","auto            "     <dn>
1970 data"s","schule          "     <gh>
1980 data"y","foto-ausg.      "     <pd>
1990 data"h","hobby           "     <km>
2000 data"q","sonstiges       "     <le>
2010 data"£","kontostand      "     <bj>
2020 data"+","summe einnahmen"      <if>
2030 data"-","summe ausgaben "      <ap>
2040 data"=","bilanz          "     <ad>
2050 data"januar   "                <jm>
2060 data"februar  "                <fb>
2070 data"maerz    "                <gg>
2080 data"april    "                <aj>
2090 data"mai      "                <dl>
2100 data"juni     "                <fg>
2110 data"juli     "                <dj>
2120 data"august   "                <bf>
2130 data"september"                <mn>
2140 data"oktober  "                <io>
2150 data"november "                <hd>
2160 data"dezember "                <lg>
2170 rem datas einlesen                  <ec>
2180 readae,aa                           <mg>
2190 fora=1to19:readkc$(a),c$(a):n
ext                                      <cd>
2200 fora=1to12:readm$(a):next:ret
urn                                      <pg>
2210 rem datas ausgeben                  <en>
2220 color1,7,3:print:printchr$(18
)chr$(29)" einnahmen"b4$b5$              <bl>
2230 b=1:fora=1toae                      <pd>
2240 ifb/2=int(b/2)thencolor1,7,4:
```

```
elsecolor1,7,6                          <hk>
2250 print" "kc$(b)chr$(18)" = "c$
(b):b=b+1:next                          <ln>
2260 color1,3,3:print:printchr$(18
)chr$(29)" ausgaben"b$                  <oi>
2270 fora=1toaa                         <ko>
2280 ifb/2=int(b/2)thencolor1,3,4:
elsecolor1,3,6                          <me>
2290 print" "kc$(b)chr$(18)" = "c$
(b):b=b+1:next                          <gc>
2300 color1,7,3:char1,1,ae+4,"":pr
intkc$(17)chr$(18)" = "c$(17)           <ob>
2310 color1,3,3:char1,1,aa+ae+6,""
:printkc$(18)chr$(18)" = "c$(18)        <pl>
2320 color1,2:char1,1,23,"":printk
c$(16)chr$(18)" = "c$(16)               <ld>
2330 color1,8,6:char1,1,22,"":prin
tkc$(19)chr$(18)" = "c$(19)             <ng>
2340 return                             <oi>
2350 rn$=chr$(18):b$=chr$(32)           <oe>
2360 b2$=b$+b$:b3$=b2$+b$               <ej>
2370 b4$=b3$+b$:b5$=b4$+b$              <cp>
2380 b$=b5$+b5$:d$=chr$(94):return      <fg>
2390 rem ========================       <pk>
2400 rem 12277 bytes memory             <le>
2410 rem 07297 bytes program            <kl>
2420 rem 00119 bytes variables          <dk>
2430 rem 01212 bytes arrays             <ap>
2440 rem 01362 bytes strings            <ki>
2450 rem 02287 bytes free (0)           <ee>
2460 rem ========================       <kp>
```

Punkt 2 (Monatsdisplay):
Nach Angabe von Monat und Jahr erscheinen alle Daten dieses Monats in der von der Eingabe her bekannten Form. Neben den Daten ist deren prozentuale Verteilung in Bezug zur Einnahme- oder Ausgabensumme angezeigt. Die „Bilanz" stellt die Differenz zwischen beiden Summen dar.
Die Wahl der Zahlen Eins und Zwei im Untermenü bewirken einen erneuten monatsbezogenen Datenabruf. Sie können hierdurch Daten zweier Monate parallel darstellen und vergleichen.

Punkt 3 (Postendisplay):
Zunächst werden Ihnen alle Einzelposten zur Auswahl gestellt.
Nach Wahl eines Kennbuchstabens und nach Angabe des betreffenden Jahres werden alle Daten des gewählten Postens in Bezug auf die einzelnen Monate des gewählten Jahres dargestellt.
Wie beim Monatsdisplay können Sie diese Daten mittels der Untermenüpunkte eins und zwei andern Daten eines beliebigen Jahres gegenüberstellen.

Punkt 4 (Notizendisplay):
Durch diese Funktion können Ihre monatsbezogenen Notizen abgerufen und dargestellt werden.

Punkt 5 (Jahresübersicht):
Alle Daten eines bestimmten Jahres werden in prozentualer Verteilung zur Einnahme- und Ausgabensumme angezeigt.
Links stehen die Kennbuchstaben der Einzelposten, unten die Kürzel der Monatsbezeichnungen.
Vor Gebrauch der Budget-Analyse ist ein Probelauf empfehlenswert.

Peter Berger □

Finanzminister

Fortsetzung von Seite 56

eine zugehörige Zahl mit maximal fünf Stellen eingeben (plus RETURN).
Falls Sie sich zu dem betreffenden Monat Notizen machen wollen (etwa Garantiezeit, Ratenzahlungen, Sonderausgaben), haben Sie nach Wahl des Notiz-Kennzeichens dazu Gelegenheit. Sie können sich pro Monat bis zu zehn Notizen machen.
Die Wahl des Abschluß-Kennzeichens bewirkt das Zusammenrechnen und die Darstellung der Daten. Der Abschluß führt auch dazu, daß die eingegebenen Daten während der Menü-Rückkehr abgespeichert werden.
Innerhalb der Eingabe-Funktion ist es nicht notwendig, alle Daten eines Monats auf einmal einzugeben, da nach Wahl dieser Funktion alle bisher zu diesem Monat/Jahr gespeicherten Daten einbezogen werden. Ergänzungen sind jederzeit möglich und Sie können durchaus täglich oder wöchentlich eingeben. Auf gleiche Weise lassen sich Irrtümer durch Eingabe von Minuszahlen korrigieren. Generelle Fehleingaben sind mit einer Löschung durch das Clear-Zeichen zu beheben, was nur den angewählten Monat beeinflußt.
Zur Darstellung der gespeicherten Daten stehen Ihnen vier Menüpunkte zur Verfügung.

SO PROGRAMMIERE ICH IN BASIC

Strategiespiel

Während die letzte Folge unserer BASIC-Reihe allgemeine Überlegungen und empfehlenswerte Vorgehensweisen bei der Programmierung behandelte, geht es heute um die Frage, wie sich eigene Spielideen entwickeln und realisieren lassen.

Obwohl es auch möglich ist, waschechte Actionprogramme in BASIC zu schreiben, ist das weite Feld der Strategiespiele wesentlich besser dazu geeignet, da die typischen BASIC-Nachteile (siehe erste Folge) hier weniger zum Tragen kommen.
Strategiespiele sind darauf ausgerichtet, den Spieler intellektuell zu fordern. Dies ist bei klassischen Brettspielen wie Schach, Halma, Mühle, Dame, Reversi, bei Legespielen wie Domino, Patience, Mah Jongg oder Geduldspielen wie Rubic's Cube der Fall.
Im weiteren Sinne kommen auch Spiele wie Monopoly, Börse oder Memory in Betracht.
Kommerzielle Computerspiele zielen meist darauf ab, den fehlenden Spielpartner durch den Computer zu ersetzen. Doch so ein Unterfangen würde die Programmierkünste eines normalen Sterblichen bei weitem überfordern, und BASIC wäre nicht unbedingt die geeignetste Programmiersprache.
Trotzdem gibt es auf dem Gebiet der Strategiespiele für den BASIC-Programmierer mehr Möglichkeiten, als man zunächst annimmt. Folgende Überlegungen machen das deutlich:
Ein geübter Schachspieler wird beim Nachspielen berühmter Partien schon oft den Wunsch gehabt haben, nach dem zwölften oder 13. Zug ganz andere Wege gegangen zu sein. Aber weiß er noch, wie bei diesem Zug die Figuren gestanden haben? Mit einem einfachen BASIC-Programm kann dem Manne geholfen werden. Zwar können Sie einwenden, daß jedes mittelmäßige Schachprogramm diese Funktion als Standard habe; Sie werden aber Mühe haben, etwas Adäquates für aktuelle oder unpopuläre Spiele zu finden.
So gerne Sie auch Monopoly oder Börse spielen mögen, die lästige Rechnerei kann alles vermiesen. Es dürfte eine Sache von zwei bis drei Abenden sein, ein perfektes Monopoly-Buchhaltungs-Programm auf die Beine zu stellen. Komplizierte, aber reizvolle Finanzspiele bekommen durch solche Hilfsprogramme einen ungeahnten Komfort.
Konventionelle Strategiespiele sind logischerweise an bestimmte Materialien wie Spielbretter, Spielsteine, Karten oder sonstige Hilfsmittel gebunden; Ihr Computer dagegen nur an Ihre Fantasie. Das bedeutet, Dinge programmieren zu können, die in der Realität nicht herstellbar wären oder deren Handhabung zu umständlich wäre.
Der Zauberwürfel als Hexaeder dürfte als BASIC-Gebilde leicht machbar sein, würde aber jeden Spielzeug-Konstrukteur an den Rand der Verzweiflung treiben. Mühle in drei Etagen zu spielen, ist auf dem C16 das reine Vergnügen, erweist sich aber als Plexiglasmodell als Flop wegen der umständlichen Handhabung.
Sie können den Gedanken noch weiter spinnen. Nichts hält Sie davon ab, „lebendige" Spielbretter zu konstruieren, die sich gemäß den Spielzügen verändern oder tückische, unsichtbare Fallen enthalten. Auch Spielfiguren können im Computer mühelos während des Spielverlaufs andere Eigenschaften annehmen. All diese Möglichkeiten sind außerhalb des Computers undurchführbar und warten geradezu darauf, von Ihnen auf dem C16 verwirklicht zu werden.
Bekannte Spiele wie Schach oder Dame können mit dem Computer auf interessante Weise verändert werden. Zum Beispiel ließe sich Schach perspektivisch in BASIC programmieren, wobei nur die Figuren dirigierbar wären, die aus der Sicht des Königs verfügbar sind.
Trotzdem gibt es relativ wenig neue Strategiespiele für Ihren Computer zu kaufen. Ein Grund kann sein, daß gute Spielideen auf dem Softwaremarkt recht rar sind und dieser Faktor bei Strategiespielen stärker zum Tragen kommt als bei allen anderen Gattungen. Nebenbei erwähnt, waren Software-Zeitschriften schon immer ein beliebtes Forum für ungewöhnliche und reizvolle Strategiespiele.

DER DREH MIT DEM WÜRFEL

Viele dieser Gesichtspunkte werden Sie in unserem Beispielprogramm PAINTER wiederfinden:

- PAINTER ist ein Gedulds- oder Strategiespiel für den Einzelspieler, dessen Spielmotivation in der Aufgabenstellung begründet ist. Es braucht also keinen computergesteuerten Gegenspieler.
- Die Spielfläche ist nur mit Computer realisierbar.
- Der Spielsinn ist dadurch gegeben, daß sich die Spielfläche während der Züge verändert.

Anhand dieses Programms können Sie unter anderem lernen, wie Sie Joysticks programmieren und dimensionierte Variablen handhaben und nutzen können. Zudem soll im Anhang gezeigt werden, wie Sie kleine Melodien selbst programmieren und in Ihre Programme einbauen.
Die Grundidee zu diesem Spiel entwickelte sich beim Betrachten mehrerer Würfel, die zufällig nebeneinander standen. Ich stellte es mir als reizvoll vor, ein Würfelfeld so zu manipulieren, daß ein darüberwandernder Stein die Würfel in Zugrichtung drehen würde; mit der Spielaufgabe, daß nach möglichst wenigen Zügen alle Würfel-Oberseiten die gleiche Augenzahl haben müßten. Das ließ sich im kleinen Rahmen mit neun Würfeln ausprobieren.
Als störend erwies sich aber, daß die Unterseite nie sichtbar war.
Doch habe ich es ja gar nicht nötig, bei meinem C16 von sechs Seiten auszugehen, es könnten ebenso fünf sein. Wenn ich statt Augen Farben nehme und meine Spielfelder so konstruiere, daß ich die Oberseite groß und hell, die angrenzenden Seitenflächen dagegen schmal und dunkel darstelle, müßte die Sache spielbar sein.
Meine Idee in Stichworten:

- Ein Spielfeld aus viermal sechs Einzelfeldern.
- Jedes Einzelfeld enthält fünf verschiedene Farben, die auf eine Oberfläche und vier angrenzende Seitenflächen verteilt sind.
- Gespielt wird mit einem Cursor, der mit dem Joystick verschiebbar über die Einzelfelder wandern kann.
- Gelangt der Cursor auf ein Einzelfeld, so wird

TONDAUERTABELLE

Note	Pause	Bezeichnung	Werte langsam	Werte schnell
		= ganze –	80	60
		= halbe –	40	30
		= viertel –	20	15
		= achtel –	10	8
		= sechzehntel –	5	4

VARIABLENLISTE:

Dimensionierte Variablen:

F% (10)	= Werte der Farbtabelle
FP% (10)	= Vertikale Positionen innerhalb der Farbtabelle
S% (5,6,4)	= Spielfeld (Farbwerte)
	S%(1,H,V) = Obere Seitenfläche
	S%(2,H,V) = Rechte Seitenfläche
	S%(3,H,V) = Untere Seitenfläche
	S%(4,H,V) = Linke Seitenfläche
	S%(5,H,V) = Oberfläche

Undimensionierte Variablen:

A,B	= Universelle Variablen
H	= Horizontale Spielfeldposition (rechnerisch)
HG	= Horizontale Spielfeldposition (grafisch)
HS	= Highscore
P	= Punkte
S1, S2, S3	= Speichervariablen von S%
SR	= Speichervariable von S%
V	= Vertikale Spielfeldposition (rechnerisch)
VG	= Vertikale Spielfeldposition (grafisch)
X	= Zufallszahl
Q1$	= Rvs ON
Q2$	= Rvs OFF

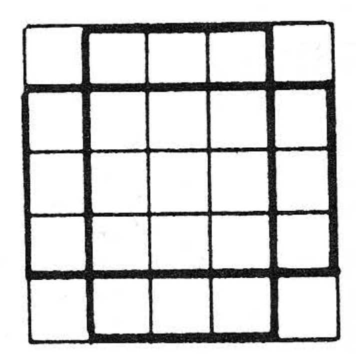
die Seitenfarbe der Zugrichtung zur Oberflächenfarbe; die Oberflächenfarbe wird zur gegenüberliegenden Seitenfarbe. Die gegenüberliegende Seitenfarbe wird zur Seitenfarbe der Zugrichtung (Rotation).

- Es soll mit möglichst wenigen Zügen erreicht werden, daß alle Oberflächen gleichfarbig sind.
- Zur Kontrolle werden die Farbcodes aller Oberflächen zahlenmäßig angezeigt.

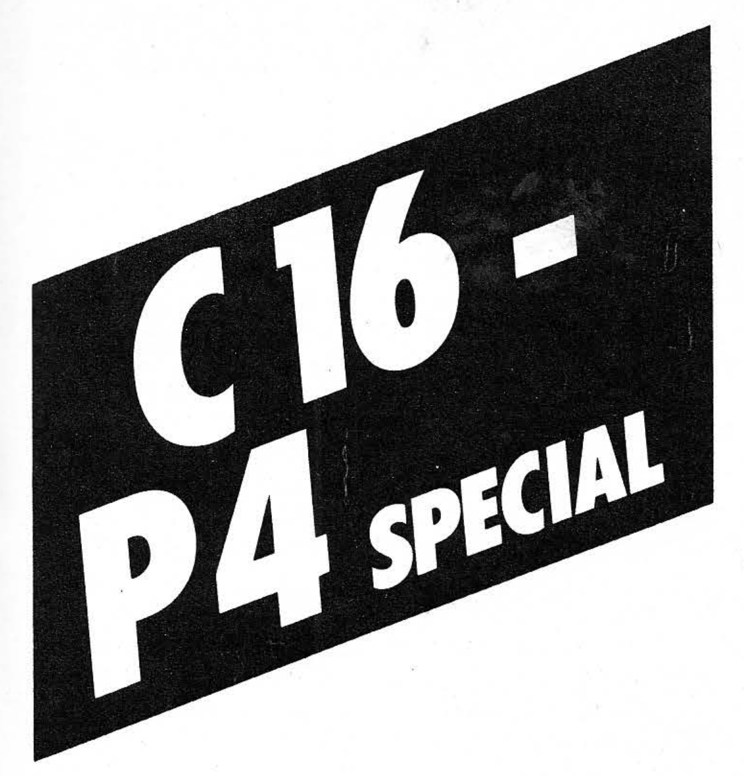
Folgende Unterprogramme habe ich vorgesehen:

9500 Ende
9100 Ausgangswerte für neue Spielrunde
9000 Ausgangswerte für Programmbeginn
8200 Zuweisung der Farben
8000 Spielfeldaufbau
5200 Highscore-Anzeige
5100 Punktanzeige
5000 Farbtabelle
4000 Einzelfelddarstellung
2100 Cursor löschen
2000 Cursor setzen
1000 Routinen zur Verschiebung der Farben
300 Joystickabfrage

Diese Aufteilung ist recht willkürlich und könnte auch anders aussehen. Für mich war es nur wichtig, daß die oft benötigten Unterprogramme (5100, 5000, 4000, 2100, 2000 und 1000) möglichst nah an der Stelle sind, von der sie am häufigsten abgerufen werden. In diesem Fall sind es Joystickabfrage und Verschiebungsroutinen.
Eine Bildschirmskizze erübrigt sich. Da ich 25 mal 40 Felder zur Verfügung habe, gestalte ich die Einzelfelder des Spiels so, daß die Seitenflächen die Breite eines Buchstabens, die Oberflächen die Breite dreier Buchstaben haben, wodurch folgende Form zustande kommt:



Jedes Einzelfeld nimmt also auf dem Bildschirm fünf mal fünf Buchstabenplätze in Anspruch. Unter Berücksichtigung seitlich plazierter Anzeigen und einer untenstehenden Kurzanleitung ergibt sich ein Spielfeld aus sechs mal vier Einzelfeldern.
Um die angrenzenden Seitenflächen separat zu halten, griff ich auf den Grafikzeichen-Vorrat des C16 zurück:



KOMMT REGELMÄSSIG ZU IHNEN INS HAUS

Finden Sie Ihr C16/P4 nicht am Kiosk? Weil es schon ausverkauft ist? Oder „Ihr" Kiosk nicht beliefert wurde? Kein Problem! Für ganze 70 DM liefern wir Ihnen per Post sechs Hefte ins Haus (Ausland 80 DM). Einfach den Bestellschein ausschneiden – fotokopieren oder abschreiben, in einen Briefumschlag und ab per Post (Achtung: Porto nicht vergessen). C16/P4-SPECIAL kommt dann pünktlich ins Haus.

WICHTIGE RECHTLICHE GARANTIE!
Sie können diesen Abo-Auftrag binnen einer Woche nach Eingang der Abo-Bestätigung durch den Verlag widerrufen – Postkarte genügt. Zur Wahrung der Frist genügt die rechtzeitige Absendung. Ansonsten läuft dieser Auftrag jeweils für sechs Ausgaben, wenn ihm nicht vier Wochen vor Ablauf widersprochen wird, weiter.

DAS SONDERANGEBOT: PRIVATE KLEINANZEIGEN KOSTENLOS!

Das bietet Ihnen COMMODORE-WELT: KLEINANZEIGEN SIND KOSTENLOSE FÜR PRIVATANBIETER! Suchen Sie etwas, haben Sie etwas zu verkaufen, zu tauschen, wollen Sie einen Club gründen? Coupon ausfüllen, auf Postkarte kleben oder in Briefumschlag stecken und abschicken. So einfach geht das. Wollen Sie das Heft nicht zerschneiden, können Sie den Coupon auch fotokopieren. Oder einfach den Anzeigentext uns so schicken, auf Postkarte oder im Brief. Aber bitte mit Druckbuchstaben oder in Schreibmaschinenschrift!
Und: Einschließlich Ihrer Adresse und/oder Telefonnummer sollten acht Zeilen à 28 Anschläge nicht überschritten werden.
ACHTUNG: WICHTIGER HINWEIS!
Wir veröffentlichen nur Kleinanzeigen privater Inserenten kostenlos. Gewerbliche Anzeigen kosten pro Zeile 4,80 plus Mehrwertsteuer!
Wir versenden für Privat-Inserenten keine Beleg-Exemplare!
Chiffre-Anzeigen sind nicht gestattet! Wir behalten uns vor, Anzeigen, die gegen rechtliche, sittliche oder sonstige Gebote verstoßen, abzulehnen!
Anzeigenabdruck in der Reihenfolge ihres Eingangs, kein Rechtsanspruch auf den Abdruck in der nächsten Ausgabe!
Die Insertion ist nicht vom Kauf des Heftes abhängig!
Wir behalten uns vor, Anzeigen, die nicht zum Themenkreis des Heftes – Computer – gehören, nicht abzudrucken oder sie nur insoweit zu berücksichtigen, wie es der Umfang des kostenlosen Anzeigenteils zuläßt.

PROGRAMMSERVICE

Hiermit bestelle ich in Kenntnis Ihrer Verkaufsbedingungen die Listings dieses Heftes auf

☐ Diskette (30 DM) ☐ **Kassetten zu 40,-**

Name ____________________

Vorname ____________________

Straße/Hausnr. ____________________

PLZ/Ort ____________________

Ich bezahle:

☐ per beiliegendem Verrechnungsscheck / Bargeld

☐ bargeldlos per Bankeinzug von meinem Konto (nur möglich in der Bundesrepublik!)

bei (Bank) und Ort ____________________

Kontonummer ____________________

Bankleitzahl ____________________

(steht auf jedem Kontoauszug)

Verkaufsbedingungen: Lieferung nur gegen Vorkasse oder Bankabbuchung. Keine Nachnahme. Umtausch bei Nichtfunktionieren.

Unterschrift ____________________

Bitte ausschneiden und einsenden an

COMMODORE WELT
KASSETTENSERVICE **16/IV**
POSTFACH 1161
D-8044 UNTERSCHLEISSHEIM

ANZEIGENSERVICE

Die große Börse für jeden Zweck in der C16/P4-SPECIAL. Kostenlos für Privat-Inserenten. Spottbillig für gewerbliche Anbieter. Einfach Coupon ausschneiden, fotokopieren o.ä., ausfüllen und ab die Post – Freimachen nicht vergessen! – Unsere Adresse steht auf dem Coupon, ebenso die Preise für gewerbliche Anbieter! Achtung! Wir weisen ausdrücklich darauf hin, daß wir offensichtlich gewerbliche Anzeigen nicht kostenlos veröffentlichen und uns jedweden Abdruck kostenloser Anzeigen vorbehalten müssen, insbesondere, wenn deren Inhalt nicht CBM-typisch ist oder gegen geltendes Recht verstößt. Private Chiffreanzeigen werden nicht aufgenommen. Für gewerbliche Anbieter DM 4,80 pro Zeile zuzüglich gesetzlicher Mehrwertsteuer.

COMMODORE WELT
ANZEIGENABTLG
POSTFACH 1161
D-8044 UNTERSCHLEISSHEIM

Name ____________________

Vorname ____________________

Straße/Hausnr. ____________________

Plz/Ort ____________________

LOAD&RUN



Das Spiele-Magazin

Hallo, liebe Leser,

Die Redaktion von Load & Run wurde mal wieder vom Fieber befallen. Kaum waren die Epidemien Shanghai und Tetris überstanden, machte sich eine neue Krankheit breit, der sich keiner der Redakteure erwehren konnte. Ich spreche vom neuesten Amiga-Game: ‚Ports of Call'. Dieser Handelsimulation ist für kurze Zeit sogar unser Verleger zum Opfer gefallen. Leider hat diese Krankheit aber auch einen Haken: Ist ein Spieler lange genug infiziert, so verflüchtigt sie sich wieder mit einem Schlag. Nach langem Spiel deckte unser Verleger einige sehr bedeutende Schwächen und Schlampereien in der Programmierung auf, die den Spielwitz erheblich senken oder gar völlig verschwinden lassen. So wurde durch schlampige Arbeit einmal mehr ein hervorragendes Spiel kaputtgemacht. Doch mehr dazu im ausführlichen Testbericht in dieser Ausgabe.

Ansonsten finden Sie in dieser Ausgabe wieder alles wie gewohnt vor. Ausführliche Spieletests berichten über die neuesten Games, Kurzberichte behandeln die Umsetzungen und Neuerscheinungen auf dem Spielemarkt, und die Player's Pages geben Hilfestellung für jeden verzweifelten Joystick-Akrobaten. Ich hoffe, es ist für jeden etwas dabei.

Viel Spaß beim Lesen wünscht Ihnen die Redaktion

Impressum

LOAD & RUN erscheint in der München Aktuell Verlags GmbH, Heßstraße 90, 8000 München 40.

Verantwortlich für den Inhalt:
Gert Seidel

Redaktion:
Thomas Bosch, Michael Nebauer.

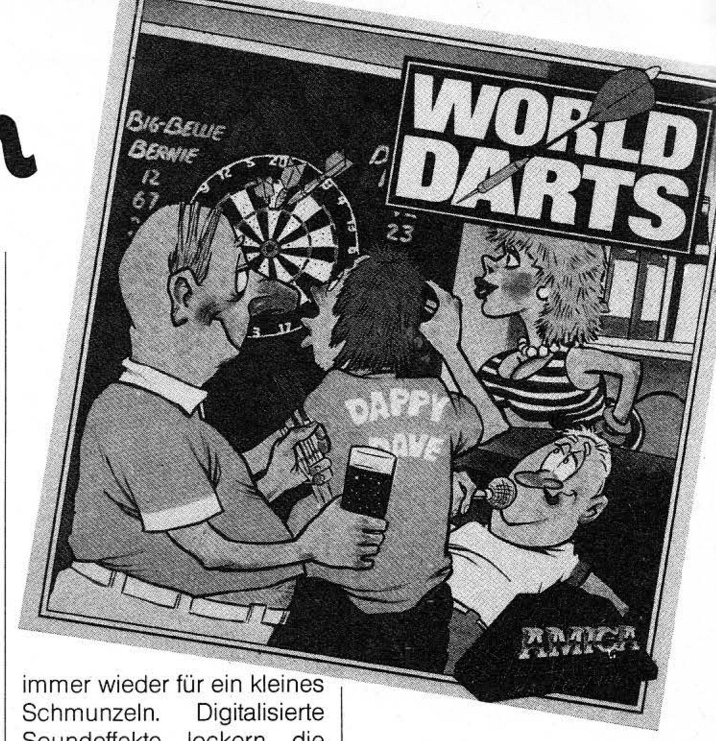
Layout:
Sonja Anderle

Anzeigenverwaltung:
ADV-Mediendienste, Aindlingerstr. 8900 Augsburg, Tel. 0821/790 42 43

Inhalt

Darts auf dem Amiga

Jetzt brauchen auch die Amiga-Besitzer nicht länger zu trauern. Mastertronic bringt in diesen Tagen eine Darts-Simulation auf den Markt.



Beim Darts-Spiel müssen Sie durch geschicktes Pfeilwerfen, im Volksmund auch „Spikkern" genannt, Ihren Punkteabstand so schnell wie möglich auf Null bringen.
Bis zu zwei Spieler können an der Darts-Simulation von Mastertronic teilnehmen. Wenn man mit Werfen an der Reihe ist, erscheint auf dem Bildschirm die Darts-Scheibe und eine Hand, welche einen Pfeil hält und leider ständig hin und her wackelt (fast wie in der Realität). Mit dem Joystick müssen Sie die Hand in die richtige Position bringen und anschließend mit dem Feuerknopf den Pfeil werfen. Danach schreibt eine weitere Hand Ihren aktuellen Punktestand auf die nebenstehende Tafel. Sieger ist derjenige Spieler, der zuerst bei Null angelangt ist.
Grafisch ist World Darts sehr gut gemacht. Viele kleine Details wie zum Beispiel wechselnde Gesichtszüge einer passiven Spielfigur sorgen immer wieder für ein kleines Schmunzeln. Digitalisierte Soundeffekte lockern die Atmosphäre zusätzlich auf. Wer sich für den Darts-Sport begeistert und zudem noch einen Amiga mit mindestens 512 Kbyte RAM und einen Farbmonitor besitzt, kann sich World Darts ja mal ansehen. TB ■

Titel:	**Darts**	
Getestet:	**Amiga**	
Umsetzungen:	---	
Im Test: ☒ Diskette ☐ Kassette	**Preis (DM):** k.A.	☒ **Joystick** ☐ **Tastatur** ☐ **Maus**

Wertung	0	25	50	75	100
Grafik					
Sound					
Bedienung					
Motivation					

Rockford is still alive

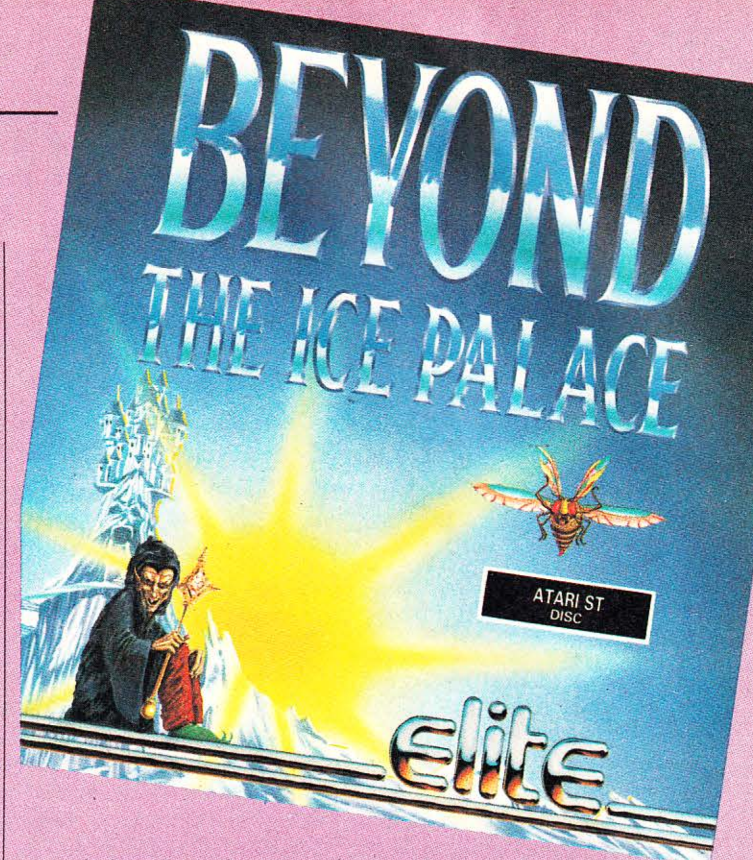
Die Boulder-Dash-Reihe reißt nicht ab. Auch für den Amiga gibt es jetzt ein solches Game.

Eine hervorragende Adaption der beliebten Boulder-Dash-Reihe liegt jetzt auch für den Amiga vor. Das Spielprinzip ist dasselbe geblieben. Rockford besitzt aber im Gegensatz zu seinen Vorgängern viele neue Features, die das Spielgeschehen auflockern. So können Sie sich zu Beginn des Spieles eine von fünf Rockford-Figuren aussuchen. Je nach Spielfigur ändert sich der Aufbau des Spielfeldes. Haben Sie sich beispielsweise für Rockford, den Meisterkoch entschieden, finden sie sich in einem Chaos aus Bestecken, Tellern und Tischen wieder. Glauben Sie aber nicht, daß sich am Spiel etwas ändert, denn auch hier müssen Sie Objekte sammeln und dabei aufpassen, daß Sie nicht von herunterfallenden Suppenschüsseln zerschmettert werden.
Jedes Spielfeld ist in vier Level unterteilt, die mit steigender Nummer schwieriger werden. Unter anderem nimmt die zur Verfügung stehende Zeit immer schneller ab.
Rockford ist grafisch sehr gut gelungen. Auch über mangelnde Sounduntermalung kann sich der Spieler nicht beklagen. Das Spiel kann ich jedem empfehlen, der einen Amiga mit Farbmonitor besitzt und noch nicht genug von der Boulder-Dash-Welle hat. TB ■

Spiele

Alte Idee – gutes Spiel

Elite, der Spezialist in Sachen Spielautomaten-Umsetzungen, hat wieder zugeschlagen. Ein gut aufgemachtes Ballerspiel für den ST ist das Ergebnis.



Lange Zeit war es still um den englischen Softwarehersteller Elite-Systems Limited, doch jetzt endlich gibt es ein neues Spiel, erhältlich für den C 64 und den ST. Die Spielidee kann man nicht mehr als alt bezeichnen, uralt trifft wohl eher zu: Ein unbekanntes Land wird von den Mächten des Bösen unterdrückt. Nach heftigen Diskussionen beschließt man, einen einsamen Helden, der zufällig über neun Leben verfügt, hinab in die Dungeons zu schicken, damit er den Zauberern und ihren Monstergehilfen ein für allemal den Garaus macht: Es darf geballert werden.

Trotz der etwas einfallslosen Spielidee dürfen Sie aber nicht glauben, daß auch das Game selbst verschrottet gehöre – im Gegenteil! Beyond The Ice Palace ist ein gutes bis sehr gutes Jump-and-run-Spiel mit hervorragender Grafik und zahlreichen Bildschirmen. Ein für ST-Verhältnisse sehr ordentliches Scrolling und gelungen animierte Sprites sorgen für einen flüssigen Spielablauf. Auch die passenden Soundeffekte lockern die Atmosphäre stark auf.

Auf dem Weg durch die Dungeons sind neben den Gegnern auch zahlreiche Hindernisse wie tiefe Schluchten oder steile Berghänge zu überwinden. Die neun Leben Ihrer Spielfigur sind wirklich nicht zu viel.

Beyond The Ice Palace ist für Freunde und Liebhaber der Jump-and-run-Spiele ein absolutes Muß. Wer noch nicht allzu viele davon in einer Sammlung hat, sollte sich das Game so schnell wie möglich zulegen. Die Atari-ST-Version benötigt übrigens ROM-TOS und einen Farbmonitor oder TV-Modulator.

TB ■

Titel: Beyond The Ice Palace
Getestet: Atari ST
Umsetzungen: ---

Im Test: [X] Diskette [ ] Kassette
Preis (DM): 59,-
[X] Joystick
[ ] Tastatur
[ ] Maus

Wertung	0	25	50	75	100
Grafik					
Sound					
Bedienung					
Motivation					

Feuer frei

Gute Nachrichten für alle Amiga-Besitzer: Wer vor einem Jahr noch mit neidischen Blicken auf Westbank für den CPC geschaut hat, darf dieses Game jetzt auch auf seinem Amiga spielen. Gunshoot ist da.

Axxion hat sich der Umsetzung von Westbank angenommen. Und weil die Amiga-Version noch ein paar Features mehr bot als die des CPC, hat man ihr gleich einen neuen Namen verpaßt: Gunshoot, was übersetzt etwa Revolverschuß heißt. Für Nicht-Kenner von Westbank hier kurz die Handlung:

Sie sind ein kleiner Bankangestellter und möchten gerne ein Held sein. Deswegen kommt Ihnen auch der Umstand, daß in letzter Zeit ziemlich viele Banken überfallen wurden, gerade recht. Mit einem 45er Revolver bewaffnet, warten Sie hinter Ihrem Schalter auf Banditen. Darum herum sind zwölf Türen, die Sie ständig im Auge behalten müssen. Von Zeit zu Zeit öffnen sich eine oder zwei und Personen betreten die Bank. Sind diese aber in der Absicht gekommen, ihr Vermögen anzulegen oder wollen sie gar die Bank um ein paar Tausender erleichtern?

Keine Chance für Bankräuber

Beobachten Sie die Leute genau. Es kann sehr schnell passieren, daß plötzlich einer seinen Colt zieht. Manche kommen sogar schon be-

waffnet herein. Sie müssen augenblicklich reagieren, wenn Sie einen Überfall vereiteln wollen. Dazu bewegen Sie mit dem Joystick ein Fadenkreuz auf die jeweilige Person. Mit dem Feuerknopf wird der Abzug betätigt. Haben Sie einen Räuber erwischt, gibt's Punkte en masse. Wenn Sie aber auf einen ehrlichen Kunden ballern, gibt's jede Menge Ärger mit dem Sheriff. Also genau hinsehen, bevor Sie abdrücken.
Gunshoot ist ein Game, das jedem zu empfehlen ist. Die ohnehin relativ hohe Spielmotivation wird durch hervorragende Grafik und digitalisierte Soundeffekte noch gesteigert. Und wenn's trotzdem mal zu langweilig werden sollte, bietet das Spiel noch einen speziellen Zwei-Player-Mode, in dem Spieler eins das Fadenkreuz steuert und sein Partner den Abzug betätigt. Gunshoot gibt's für alle Amigas ab 512 KByte RAM mit Farbmonitor. TB ■

Titel:	Gunshoot	
Getestet:	Amiga	
Umsetzungen:	---	
Im Test:	Preis (DM):	
☒ Diskette	k.A.	☒ Joystick
☐ Kassette		☐ Tastatur
		☐ Maus

Wertung	0	25	50	75	100
Grafik					
Sound					
Bedienung					
Motivation					

Vorsicht: Suchtgefahr!

Ein einfaches Puzzle-Spiel – das suggeriert die Aufmachung von Powerstyx. Doch hinter diesem Namen verbirgt sich sehr viel mehr.

In Powerstyx müssen Sie verschiedene Bilder freilegen, die zu Beginn noch unsichtbar sind. Mit einem Cursor, der nur mit dem Joystick gesteuert werden kann, schneiden Sie aus dem Bildschirm beliebig große Rechtecke heraus. Dadurch wird ein Teil des verdeckten Bildes sichtbar, der jeweils so groß ist wie das ausgeschnittene Rechteck. Sind mehr als 75 Prozent des Bildes freigelegt, gelangen Sie in den nächsten Level. Es gibt 15 Bilder zu entdecken, jedes repräsentiert einen Level.
Mit dem Cursor Rechtecke ausschneiden – das klingt doch wirklich nicht schwierig. Und tatsächlich weckt das Game zunächst keine große Begeisterung. Mißmutig bewege ich meinen Cursor über den Bildschirm. Hier ein Rechteck, da ein Rechteck. Doch was ist das? Aus der linken oberen Ecke tauchen plötzlich zwei Totenköpfe auf, die meinen Linien folgen. Da die beiden nicht gerade vertrauenerweckend aussehen, entschließe ich mich zur Flucht nach hinten. Doch diverse fliegende Buchstaben versperren mir den Weg. Nach wenigen Sekunden gebe ich den Kampf auf – und bin eines meiner fünf Leben los.
Der zweite Anlauf gelingt endlich. Nach knapp vier Minuten, die zu 80 Prozent aus Flucht vor diversen bösartigen Sprites bestehen, ist das erste Bild freigelegt oder besser gesagt: 75 Prozent davon. Das Bild wird komplett aufgebaut und ich kann mich knappe 30 Sekunden lang auf die wirklich einmalig schöne Grafik konzentrieren. Noch bevor ich mich richtig sattgesehen habe, ertönt ein akustisches Warnsignal und der Bildschirm wird wieder schwarz, was den nächsten Level ankündigt. Nun gut, ich habe den ersten in nur vier Minuten überstanden, da wirft mich der zweite auch nicht um – denkste! Die fliegenden Totenschädel haben Verstärkung bekommen und auch die restlichen Sprites scheinen wesentlich angriffslustiger. Nach drei Minuten muß ich kapitulieren, da meine Leben aufgebraucht sind. Ein neuer Start, ein neues Spiel.

Untermalt wird das Spektakel von diversen Hintergrundgeräuschen, die nicht sonderlich gut gelungen sind. Das fällt aber nicht allzu stark ins Gewicht, da sich der Spieler ständig auf das Spielgeschehen konzentrieren muß. Bis in den 15. Level habe ich es zwar nicht geschafft, aber die hervorragende Qualität der Grafiken dürfte sich wohl kaum verschlechtern. Übrigens sind die Bilder bei der C64-Version fast bessser als die der Amiga-Version. Auch läuft auf dem C64 alles ein klein wenig schneller ab. Aber keine Angst, liebe Amiga-Freaks, auch die Geschwindigkeit auf dem 16-Bit-Computer reicht doppelt und dreifach aus.
Powerstyx sollte in keiner Sammlung fehlen. Ich kann es nur wärmstens weiterempfehlen. Doch lassen Sie sich warnen: Wer es einmal gespielt hat, kommt so schnell nicht mehr davon los.
Thomas Bosch ■



Ports of Call

Stellen Sie sich vor, Sie bekommen fünf Millionen Dollar und wollen daraus 500 Millionen Dollar machen. Bei Ports of Call kann das leicht gelingen. Wer sich allerdings keine Mühe gibt, bei dem wird bald der Pleitegeier über der Reederei schweben.



Ports of Call ist ein Spiel, bei dem der Spieler die Rolle eines Reeders übernimmt. Mit fünf Millionen Dollar müssen nach dem Spielstart erst einmal Schiffe gekauft werden. Am billigsten sind Low-Coast-Ships, diese sehen allerdings auch so aus. Die verbeulten Kähne sollten erst einmal repariert werden. Der Zustand des Schiffes wird in jedem Hafen angegeben, die Tauglichkeit sollte niemals 60% unterschreiten, da sonst das Schiff von einer Rattenplage befallen werden und später sogar untergehen kann. Ist das Schiff also gerade noch schwimmfähig, kann die Ladung ausgewählt werden. Manche Güter können sehr viel Frachterträge bringen, doch bei der Auswahl ist auf sehr viele Dinge zu achten: Wie weit ist es vom Starthafen zum Zielhafen, wieviel Treibstoff wird verbraucht, lohnen die Frachten am Zielhafen, wie hoch ist die Gefahr, Eisberge oder Riffe umfahren zu müssen, sind in diesem Gebiet viele Stürme oder drohen Freibeuter den Tanker zu plündern ? Ist die richtige Entscheidung gefallen, kann es schon losgehen. Zum Ablegen können Schlepper zu Hilfe genommen werden, diese kosten allerdings auch Geld. Außerdem ist auf die Schlepperkapitäne auch kein Verlaß, hin und wieder streiken diese und da ein Reeder keine Zeit zu verlieren hat, heißt es, selbst im Hafen ab- bzw. anzulegen. Dies hört sich zwar sehr einfach an, muß aber erst einmal ausdauernd geübt werden. Ein kleiner Crash gegen die Hafenmauer kostet nämlich viel Zeit und Geld. Sobald das beladene Schiff sicher den Hafen verlassen hat, kann der Spieler auf der Weltkarte sein kleines Schiff entlangtuckern sehen. Unterwegs kann auch so manches Ereignis geschehen, so ist es manchmal nötig, einem Sturm auszuweichen oder Schiffbrüchige zu retten. Im Zielhafen heißt es, nach neuen ertragreichen Ladungen umsehen. Ist keine gewinnbringende Fracht in Sicht, kann ja mal das Schiff auf Vordermann gebracht werden. Nach ein paar Tagen, das Schiff glänzt nun wieder in der Abendsonne, muß dann Geld verdient werden. Deshalb sollten Häfen, in denen es keine ertragreichen Güter gibt, gemieden werden.

Dunkle Geschäfte können natürlich auch getätigt werden. Für einen Schrankkoffer, der irgendwo im Schiff versteckt wird, erhält der Reeder in den meisten Fällen 100.000 Dollar. An 50 Kisten Waffen hingegen kann schon einmal eine Millon Dollar verdient werden. Nur: Der Zoll findet diese Schmugglerei nicht so toll. In den meisten Fällen genügt aber ein saftiges Schmiergeld, um das betroffene Schiff wieder freizukaufen. Sind die ersten 20 Millionen Dollar erwirtschaftet, kann der Reeder ein High-Tech-Schiff mit Querstrahlruder anzahlen. Mit 40% Anzahlung kann sich fast jede Reederei dieses Superschiff leisten. Aber aufgepaßt, mehr als 40 Millionen Dollar sollte für so einen großen Pott nicht bezahlt werden. Natürlich hat jeder Reeder auch ein Büro, von hier aus können die Gewinne und die Verluste abgefragt werden, Hypotheken getilgt oder Kredite aufgenommen werden. Falls mal irgendwann die Geschäfte nicht mehr so gut laufen, kann mit dem Menüpunkt Registerwechsel der Heimathafen des Reeders gewechselt werden. Zu den ertragreichsten Frachtrouten gehören: Rotterdam – Kalkutta und Vancouver – Karachi.

Mittlerweile wird dieses Spiel so gepriesen, als ob es noch kein besseres Wirtschaftsspiel gegeben hätte. Anfangs macht es sicherlich unwahrscheinlich viel Spaß. Doch nach einigen Tagen, wenn der Spieler genug Geld erwirtschaftet hat, verschwindet die Spieldiskette wieder in der Versenkung. Denn die Fehler, die dieses Spiel hat, machen es nach einer gewissen Zeit total lächerlich.

Das fängt schon damit an, daß ein auf Reede liegendes Schiff in Rio de Janeiro von einem Eisberg gerammt werden kann. Es kann weiter mal passieren, daß einem Spieler unterwegs der Treibstoff ausgeht. Das betroffene Schiff muß dann natürlich abgeschleppt werden.

Doch wenn im Hafen wieder aufgetankt werden soll, ist der Dieselbunker noch fast voll. Dem Milliardär macht dieses lächerliche Milliönchen Abschleppkosten überhaupt nichts aus. Doch der kleine Reeder sieht mal wieder den Pleitegeier.

Oder folgender Fall tritt ein: Wegen mangelnder Aufträge müssen die Schiffe im Hafen festgemacht werden. Diese Schiffe rühren sich dann, für die angegebene Zeit, nicht mehr vom Fleck. Doch wenn dann ein anderes Schiff den „Kurs“ des eigenen Pottes kreuzt, welches man mittels einem Radar umfahren sollte, muß man sich schon an den Kopf greifen. Das gleiche Desaster haben die armen Freibeuter, diese greifen meist erst an, wenn das Schiff leer im Hafen liegt oder sie es bereits ausgeraubt hatten. Wenn beim Einlaufen mal wieder die Schlepperkapitäne streiken sollten, ist es immer billiger, das Schiff rückwärts aus dem Hafen zu steuern, denn der Strafzettel ist allemal niedriger als die folgenden Reparaturkosten-

Rechnung. Daß ein Schiff mal auf Kollisionskurs fahren kann, ist die natürlichste Sache der Welt, aber dann dem geisterfahrenden Schiff im Rückwärtsgang zu entfliehen, ist sicher keine Lösung. Irgendwann mal kommt der Tag, an dem sich der Spieler entscheidet, 100 High-Tech-Schiffe zu kaufen. Das benötigt dann schon eine Ewigkeit, aber dann den Spieler hundertfach zu zwingen, alle seine noch im Hafen liegenden Schiffe aus dem Kollisionsbereich zu fahren, ist schon eine Zumutung. Irgendwann pfeift jeder auf Frachten, er kauft die hochmodernen Schiffe für unter ca. 40 Millionen und verkauft diese wieder für ca. 70 Millionen, ohne daß das Schiff jemals den Hafen verläßt.

Im Grunde kann man dieses Spiel niemandem empfehlen. ah ■

Freistoß für Amiga

Die Fußball-EM ist zwar vorbei, aber das bedeutet noch lange nicht, daß jetzt eine Fußball-tote Zeit folgt. Besitzer eines Atari ST oder Amiga können die EM auf ihrem Computer nachvollziehen.

Von Grandslam, bekannt durch Spiele wie Terramex oder Fred Feuerstein, kommt ein neues Sportspiel zu uns: Euro Soccer '88, eine Fußball-Simulation. Da bislang kein übermäßig großes und qualitativ hochwertiges Soccerspiel auf dem Markt ist, war ich entsprechend skeptisch. Allerdings sollte ich eine Überraschung erleben, denn Euro Soccer '88 dürfte nur schwer zu schlagen sein. Sowohl grafisch als auch vom Sound her gehört dieses Spiel zum Besten, was für die bekannten 16-Biter zu haben ist. Große detaillierte Sprites und zahlreiche grafische Effekte motivieren den Spieler.

Zu Beginn wählen Sie sich Ihre Mannschaft aus. Wenn Ihnen die Trikot-Farben nicht zusagen, können diese selbstverständlich geändert werden. Nachdem die wichtigsten Parameter eingestellt worden sind, geht's los.

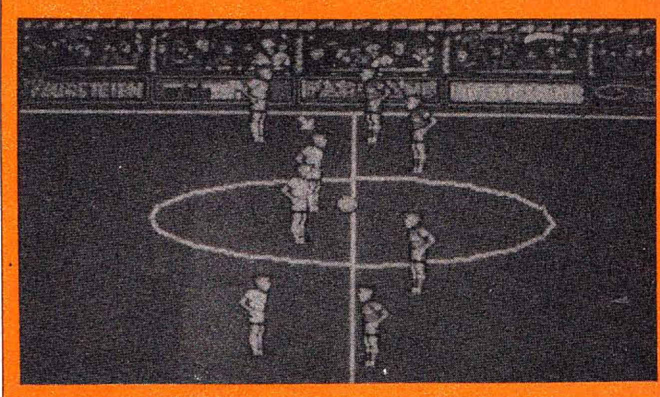
Wie von den bisherigen Fußball-Simulationen gewohnt, können Sie nur die Steuerung für jeweils einen Spieler übernehmen; welchen, wählt der Computer aus. In den meisten Fällen ist es derjenige, der dem Ball am nächsten ist. Damit Ihnen keine Verwechslung unterlaufen kann, schwebt über der aktuellen Spielfigur ein blinkender Pfeil. Auch die Figur des (Computer)-Gegners ist mit einem Pfeil gekennzeichnet.

Der oder die Spieler steuern die Fußballteams im Turnier, der Computer simuliert die Resultate der übrigen Matches. Damit ist schon in den Entscheidungsspielen Spannung und Nervenkitzel garantiert. Dann geht's ins Halbfinale und in die Endrunden.

Euro Soccer '88 wird in einer stabilen Kunststoffverpakkung auf Diskette geliefert. Als kleine Zugaben finden Sie in der Packung noch ein DIN-A2-Poster und einen Ansteck-Button. Die Anleitung ist komplett in Deutsch verfaßt, so daß auch Spieler, die der englischen Sprache nicht mächtig sind, keine Probleme haben werden.

Das Spiel kann jedem, der sich für Fußball begeistert und ein hervorragend aufgemachtes Spiel schätzt, wärmstens empfohlen werden. Den Preis von knapp 60 Mark dürfen Sie ohne Bedenken auf den Tisch legen. TB ■





Titel:	**Euro Soccer '88**
Getestet:	**Amiga**
Umsetzungen:	**Atari ST**

Im Test: ☒ Diskette ☐ Kassette

Preis (DM): k.A.

☒ Joystick ☒ Tastatur ☒ Maus

Wertung	0	25	50	75	100
Grafik					
Sound					
Bedienung					
Motivation					

Die Wargames-Edition

Vor kurzer Zeit erreichte uns eine wahre Flut Kriegsstrategie-Spiele von Electronic Arts Deutschland. Die Spielefreaks und auch die Programmierer scheinen neuerdings Gefallen an dieser zweifelhaften Art von Games zu finden.

Ich mußte lange überlegen, ehe ich mnich entschloß, die Kriegsspiel-Serie von Electronic Arts nehmen. Mit Vorliebe wird der Zweite Weltkrieg behandelt. Aber auch Kollektionen, mit dem jedes nennenswerte Gefecht durchgeführt werden kann, sind dabei. Die Palette reicht vom Flugzeugträgerkrieg im Pazifik über die Schlacht Rommel gegen Patton bis hin zum Bombardement Englands und Deutschlands.

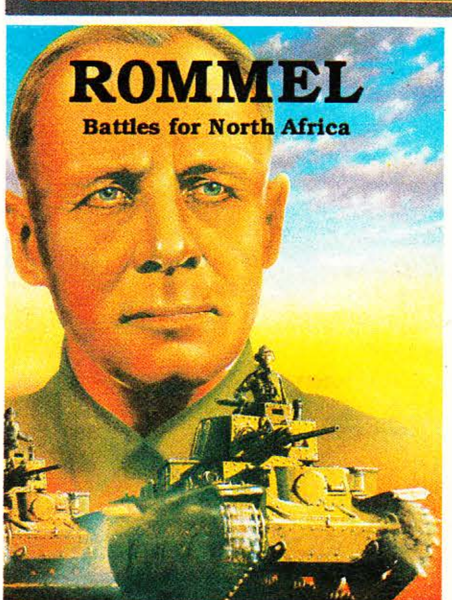


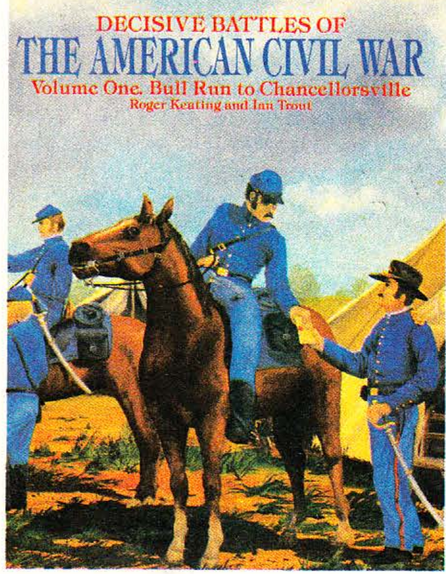
Patton versus Rommel

nennt sich das erste und zugleich beste Spiel dieser Wargames. Es simuliert das Gefecht in der Normandie, als sich der britische General Patton dem deutschen Feldmarschall Rommel gegenübersah.

Das Kriegsspektakel spielt sich im Computer nur auf einer grafisch ganz gut gelungenen Landkarte ab. Alle Einheiten erscheinen auf der Karte als gut erkennbare Symbole, die mit dem Joystick anzuwählen und zu steuern sind. Meldungen und Bestätigungen über eventuelle Aktivitäten erscheinen als gut lesbare Fenster innerhalb der Kartengrafik.

Über den Witz dieser Simulation läßt sich streiten. Ich konnte jedenfalls nur kurze Zeit vor dem Bildschirm ausharren, ehe mir die ganze Sache zu dumm wurde. Aber vielleicht gibt es doch so ausgefuchste Strategen, die dieses Game als eine Herausforderung betrachten.





Rommel – Battles for North Africa

Game zwei beschäftigt sich ebenfalls mit dem Wüstenfuchs Rommel. Bei diesem Spiel können Sie gegen den Computer oder einen zweiten Kommandeur spielen, der entweder die Rolle der Alliierten oder die von Rommel übernimmt. Hier haben Sie jedoch auch die Möglichkeit, unter mehreren Schlachten in Nordafrika zu wählen.

Auch bei diesem Game läuft die Handlung auf der Landkarte ab. Vergleiche mit dem oben genannten bei der Grafik unterlasse ich lieber. Die Darstellung der Karte ist der reinste Horror.

Durch die vielen verschiedenen Farben werden Sie so stark verwirrt, daß Sie sich auf der ohnehin schon Blockgrafik-artigen Karte kaum noch zurechtfinden. Überhaupt ist das ganze Spiel reichlich mit Farbe bestückt, oder besser gesagt: überladen.

Dafür kommt der Sound zu kurz. Wenn überhaupt, dann tönt nur ein langweiliges Gedudel aus dem Lautsprecher, das dem C-64 nicht im geringsten würdig ist.

Gehen wir jetzt etwas in der Zeit zurück, nämlich ins Jahr 1861, als in Amerika der große Ärger begann.

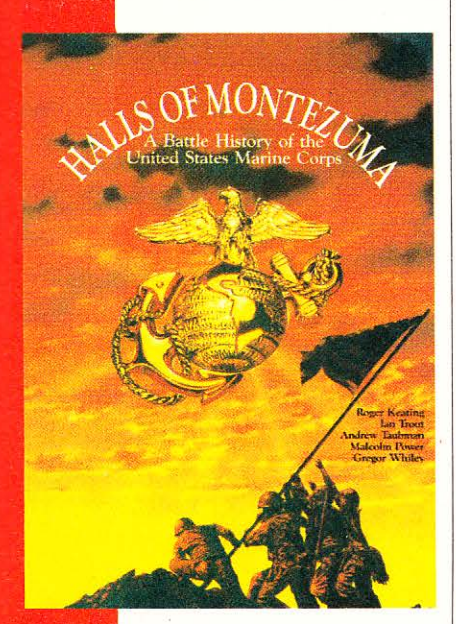
The American Civil War

Der Amerikanische Bürgerkrieg unterscheidet sich im Aufbau, zumindest auf dem Computer, fast nicht von Rommels Schlacht um Nordafrika. Der Bildschirmaufbau ist vollkommen identisch, die Farbgebung auch, und am Sound wurde ebenfalls

nichts verbessert. Einziger Unterschied ist das Szenario. Bedient wird das Spiel über die Cursortasten. Eine Joystick-Steuerung, die wesentlich komfortabler gewesen wäre, wurde nicht eingebaut. Auch hier flaut die Motivation schon nach kurzer Spieldauer ab. Hinzu kommen die langen Wartezeiten beim Laden der Disk, da oft nachgeladen werden muß. Insgesamt also ein sehr mäßiges Spiel.

Halls of Montezuma

Für ganz Kriegslüsterne kommt von denselben Programmierern ein Simulationsspiel, das mehrere Schlachten, unter anderem auch in den zwei Weltkriegen, zusammenfaßt. Viel mehr kann ich dazu nicht mehr sagen, außer daß sich die Kämpfe vom ersten Weltkrieg über den zweiten und den Koreakrieg bis hin nach Vietnam ziehen. Die ehrenvollen Gemetzel werden natürlich von der amerikanischen Elitetruppe Marines ausgetragen.



Da dieselben Designer am Werk waren, hat sich auch hier gegenüber den anderen Games nicht das geringste geändert. Die Grafik blieb gleich, der Sound auch.

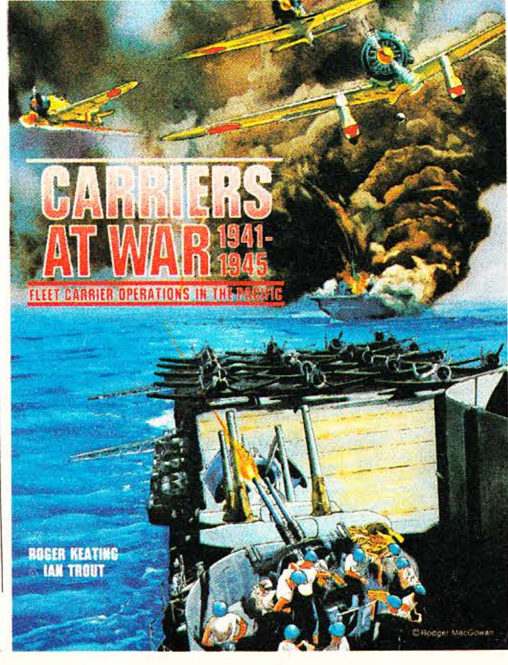
Gehen wir wieder zurück zum Zweiten Weltkrieg und sehen den Flugzeugträgern im Pazifischen Ozean zu.

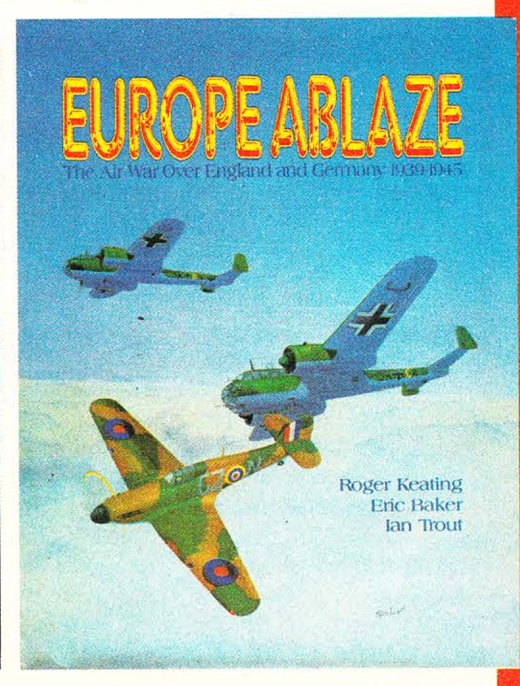
Carriers AT War 1941–1945

Es ist nicht leicht, einen richtigen Eindruck von diesem Game zu bekommen, ohne die beiden ausführlichen, sechzehn- und vierundzwanzigseitigen Handbücher komplett durchstudiert zu haben.
Ort und Zeit der Handlung ist wieder einmal der Zweite Weltkrieg im Pazifischen Ozean. Zur Auswahl stehen sechs verschiedene Schauplätze, unter anderem Pearl Harbour, Coral Sea, die Midway-Inseln und die Philippinische See. Alles Orte, die erfahrenen Veteranen wohl ein Begriff sind. Hier haben sich zwei Flugzeugträger erbitterte Schlachten geliefert: Die amerikanische „Saratoga" traf auf die japanische „Kaga".
Der Bildschirmaufbau ist wieder einmal der gleiche wie bei den obengenannten Games. Die Steuerung ebenfalls, der schlechte Sound auch.
Trotzdem scheint Carriers at War umfangreicher ausgefallen zu sein, da dem Spiel eine ganze Menge an Handbüchern und Kartenmaterial, das nicht nur bloßer Schnick-Schnack ist, beiliegen.
Kommen wir zum letzten Spiel dieser Episode: der Bombardierung Europas.

Europe Ablaze

In derselben Aufmachung wie Carriers at War geliefert, enthält auch Europe Ablaze jede Menge an Handbüchern und Karten. Diesmal findet die Handlung jedoch nicht auf hoher See statt, sondern hoch in den Lüften. Europe Ablaze simuliert den Luftkrieg über England und Deutschland.
Wie sollte es anders sein, das weitere entspricht den anderen Games. Grafik und Soundeffekte sind völlig identisch, an Komplexität erreicht es mit Leichtigkeit das Pensum von Carriers at War. Gespielt werden kann in drei Szenarien: die Schlacht um England, die Nachtoffensive britischer Bomber und der Angriff auf Berlin.
Ein Wort zu allen hier getesteten Kriegsstrategie-Spielen: Die Aufmachung und die Illustration könnte nicht besser sein. Dem Spieler werden die historischen Hintergründe und Gegebenheiten genau erklärt sowie die verwendeten Waffen, Flugzeuge, Panzer und Schiffe detailgetreu erläutert. Die Handbücher lassen keine Fragen offen und sind sorgfältig und mit viel Liebe zum Detail erarbeitet.
Trotzdem habe ich erhebliche Bedenken gegen das Thema dieser Games. Sie befassen sich leider nur – ohne Ausnahme – mit Krieg. Natürlich, Kriege sind die beste Übung für strategisches Denken, trotzdem sollte man es damit nicht übertreiben. Die Programmierer haben sich große Mühe gegeben, die Schauplätze und Situationen so realitätsgetreu wie möglich zu gestalten. Technisch gesehen sind die Games daher kaum zu übertreffen, obwohl ich das von der grafischen Ausführung nicht gerade behaupten würde.
Kurz gesagt: Es ist Geschmackssache, ob die Spiele gefallen oder nicht. Mich sprechen sie nicht besonders an. Es war außerordentlich schwierig, völlig objektiv zu bleiben, und ich gebe gerne zu, daß mir das nicht ganz gelang. Ausgefuchste Strategen sehen darin aber sicher eine große Herausforderung. Fraglich ist nur, ob sich nicht die Bundesprüfstelle bald für diese Art von Herausforderung interessiert. mn ■





Horrortrip als Comicstrip

Sie haben gerade Lust, sich mal wieder kräftig zu gruseln? Sie haben auch gute Nerven? Und Sie wollen nebenbei auch mal ganz herzhaft lachen? Dann habe ich einen guten Tip für Sie: Besuchen Sie doch mal Rue Morgue, Hausnummer 666. Bitte klingeln bei „Ooze".

Mr. Cheez Burger, wohnhaft in 4233 Denbrough 3575, Rue Morgue No. 666, ist am 28. August 82 um 10 Uhr 25 Minuten in Denbrough, Rue Morgue 666 tot aufgefunden worden.
Der gräßlich verstümmelte Leichnam, die abgetrennten Glieder und die völlige Blutleere des Körpers weisen auf einen möglicherweise rituellen Mord hin. Ebenso die Striemen auf der Haut des Ermordeten, deren Entstehung noch immer unklar bleibt, da sie tiefe Brandmale mit aufweisen. Weitere Ergebnisse der Obduktion können beim zuständigen Gerichtsarzt eingesehen werden."
So steht es in der Sterbeurkunde Ihres so merkwürdig dahingeschiedenen Onkels Cheez Burger. Als Zeugen unterzeichneten unter anderem Mr. Stephen King und Mr. Edgar Allen Poe.
Wenn Sie das gelesen haben, werden Sie sich wohl fragen, was es daran so herzlich zu lachen gibt. Richtig, hier noch gar nichts. Das war nur eine kleine stimmungsmachende Beilage zum deutschen Grafikadventure „OOZE - Als die Geister mürbe wurden."

Einfallsreiche Hintergrundgeschichte

Wenn Sie die mit einem hervorragenden Cover bedeckte Verpackung öffnen, fallen Ihnen zuerst die besagte Sterbeurkunde und ein dreißigseitiges Heft entgegen, das sich als Tagebuch Ihres verstorbenen Onkels entpuppt. Der Roman ist sehr stimmungsvoll geschrieben und eignet sich hervorragend als Einführung in das Abenteuerspiel, zumal die Geschichte dort aufhört, wo das Spiel beginnt.
Als Cheez Burger in die alte Villa einzog, bekam er zuerst große Schwierigkeiten mit den dort wohnenden Geistern aller Art. Diese waren es nämlich nicht gewöhnt, ein fremdes lebendes Wesen zu sehen, das sie möglicherweise in ihrer Ruhe stören könnte. Also bereiteten sie dem alten Cheez kurzerhand eine Menge Terror – bis der Hausgeist Ludus, ehemaliger Hofwitzbold des Königs, herausfand, daß Mr. Burger eigentlich ganz nett ist und den Geistern gar nichts anhaben will. Nach einigen skurrilen Streichen fanden die beiden endlich zusammen und wurden unzertrennliche Freunde – kein Wunder, wenn ein Teil unsterblich war. Nachdem Ludus seine Spukkollegen, T-Bone das Skelett, Vino, den ewig betrunkenen Weingeist und Holunder, dessen liebstes Hobby ist, mit Köpfen zu kegeln, davon überzeugt hatte, daß Cheez Burger zu ihnen hält, hatte dieser eine Gruppe fester Freunde gewonnen.
Leider jedoch hatten auch die Geister ein großes Problem, in das sie Ihren Onkel einweihten: Haustyrann in Carfax Abbey ist ein Monster, das sich Ooze nennt und alle anderen Geister in Schach hält. Dieser Ooze, der laut Ludus nicht mal ein wirklicher Geist ist, weil er noch nie gelebt hat, ist durch und durch böse und hat überall seine Handlanger im Haus versteckt. Ooze hat aber auch eine sehr bedeutende Stärke: Er kann seine Gestalt beliebig verändern.

Roman zum Spiel als Beigabe

Der Plan, Ooze zu vernichten, schlug fehl. Das Ergebnis können Sie in der Sterbeurkunde von Cheez Burger nachlesen.
Hier beginnt das Grafikadventure. Sie müssen als Ham Burger, der Erbe von Cheez, das Haus übernehmen und versuchen, Ooze mit all seinen Spionen zu vertreiben. Daß das gar nicht so einfach ist, hat auch schon Ihr Onkel zu spüren bekommen.
Das wichtigste an einem Abenteuerspiel ist der Eingabeinterpreter. In diesem Fall setzt der deutsche Parser wohl neue Maßstäbe. Er versteht ein großes Stück des Vokabulars und auch die deutsche Grammatik einwandfrei. Lange Sätze, durch Kommas und Bindewörter verknüpft, stellen für ihn kein Problem dar. Wenn er einmal eine Eingabe nicht versteht, dann läßt das Programm Sie das deutlich spüren, indem

es Sie durch einen frechen Kommentar darauf aufmerksam macht.
Eine kleine Schwäche hat der Parser jedoch: Schnelltipper werden sich umgewöhnen und einen langsameren Gang einlegen müssen. Tippen Sie nämlich zu schnell, so kommt der Interpreter nicht mehr mit oder läßt der Bequemlichkeit halber einfach ein paar Buchstaben aus. Nach einer kurzen Eingewöhnungszeit stellt das jedoch auch kein Problem mehr dar.
Der Textteil ist in Deutsch verfaßt und enthält nicht den kleinsten Rechtschreib- oder Grammatikfehler. Vom Umfang her stellt Ooze sogar die Infocom-Produkte in den Schatten. Man kann ohne Übertreibung sagen, daß das neue Dragonware-Game einem illustrierten Horrorroman in der Ich-Perspektive sehr nahe kommt, so detailreich und durchdacht sind die riesigen Textmassen.
Zu einem Grafikadventure gehören natürlich auch Grafiken. Denen von Ooze kann wirklich das Prädikat „Hervorragend" verliehen werden. Zu jedem der siebzig Räume gibt es eine eigene Grafik, an der ich stundenlang die Augen weiden könnte. Teilweise lassen die Bilder sogar die von Jinxter und The Pawn hinter sich.
Mit den Texten und Bildern gelang es den Programmierern ausgezeichnet, eine Horrorstimmung zu erzeugen. Am besten, Sie versuchen nachts, bei Kerzenlicht und geeigneter Musik, Ooze zu schlagen. Aber auch tagsüber läßt die Spannung nicht lange auf sich warten.

Fazit

Mit „Ooze - Als die Geister mürbe wurden" wurde das bislang beste deutsche Adventure programmiert, das sich hinter amerikanischen Produkten wie The Prawn nicht zu verstecken braucht. Durch den gelungenen Parser, die hervorragenden Texte, die sehenswerte Grafik und die witzige Vorgeschichte wird es für deutsche Abenteuerspiele neue Maßstäbe setzen, die zu übertreffen für andere Firmen schwierig sein wird.

mn ■

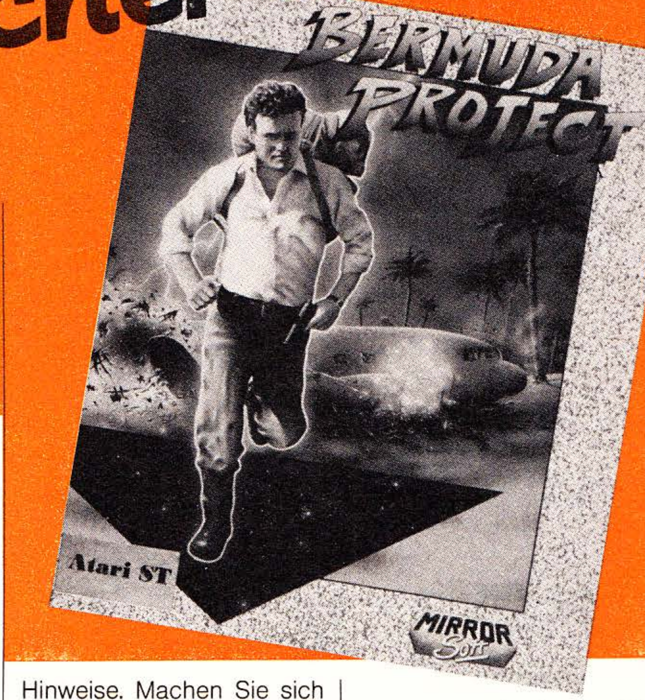
Titel:	**Ooze**
Getestet:	**Amiga**
Umsetzungen:	**Atari ST, MS-DOS, C-64**

Im Test:	Preis (DM):	
☒ Diskette	79,90	☐ Joystick
☐ Kassette		☒ Tastatur
		☐ Maus

Wertung	0	25	50	75	100
Grafik					
Sound					
Bedienung					
Motivation					

Ein mörderischer Auftrag

Schon seit über 50 Jahren häufen sich die Rätsel um verschwundene Schiffe und Flugzeuge im Bereich des Bermuda-Dreiecks. Nun soll ein Reporter Licht in die Sache bringen.



Das Rätsel, welches das Bermuda-Dreieck umgibt, beschäftigt schon lange die Wissenschaftler und Theologen. Ganze Schiffskonvois und Flugzeugkolonnen verschwinden plötzlich spurlos vom Radarschirm und tauchen nie mehr auf. Von Mirrorsoft kommt jetzt ein Computerspiel zu uns, das sich mit dem Bermuda-Mysterium beschäftigt.
Sie übernehmen die Rolle eines hochbezahlten Reporters, der von seiner Zeitung den Auftrag erhält, sich auf den Bermuda-Inseln ein wenig umzusehen. Doch auch Sie entkommen dem berüchtigten Dreieck nicht. Ihr Flugzeug stürzt ab.
Nach mehrstündiger Bewußtlosigkeit finden Sie sich auf einer offenbar unbewohnten Insel wieder. Ihr einziger Begleiter, der Pilot, ist tot. Was nun?
Wie sie weiter vorgehen, bleibt Ihnen überlassen. Auch die Anleitung gibt keine Hinweise. Machen Sie sich also auf die Suche nach Einwohnern und spüren sie mit deren Hilfe die Wracks der verschwundenen Schiffe und Flugzeuge auf. Dabei gelten die gleichen Bedingungen wie in der Realität, das heißt, Sie müssen auch für Nahrung und Trinkwasser sorgen und dürfen Ihrer

Fortsetzung auf S. 14

Spiele

Games für kleine Geldbeutel

Daß gute Software nicht unbedingt teuer sein muß, haben uns diverse Hersteller wieder einmal mit einer geballten Ladung Billigspiele bewiesen.

Alle hier getesteten Billigspiele werden in der gewohnten Kunststoffverpackung auf Kassette geliefert. Zu erhalten sind die Games für etwa zehn bis fünfzehn Mark in den Fachabteilungen der Kaufhäuser und im Computer-Fachhandel.

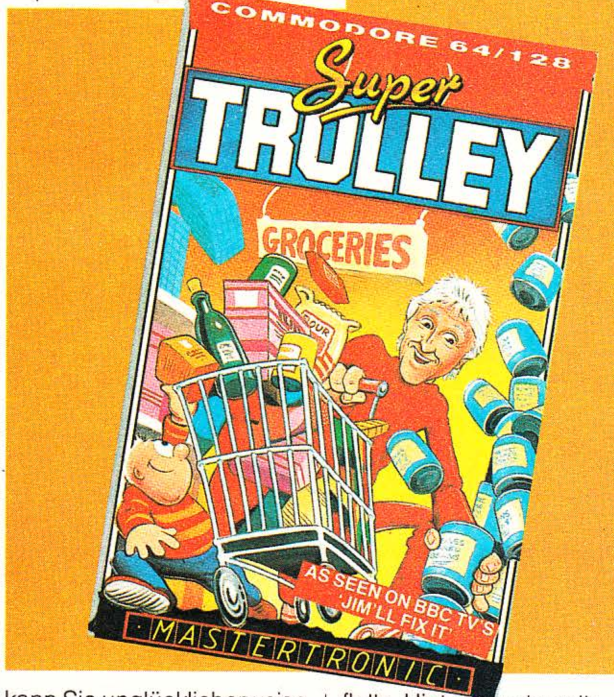
Starquake

Eine geheime Macht hat vom Universum Besitz ergriffen. Zahlreiche Bösewichte haben die friedliebenden Bewohner verjagt und treiben nun ihr zerstörerisches Unwesen. Ihre Aufgabe liegt auf der Hand: Stürzen Sie sich ins Getümmel und machen Sie den unerwünschten Eindringlingen den Garaus. Starquake, das bereits vor einem guten Jahr auf den Markt kam, ist ein Arcade-Action-Game für einen Spieler. Mastertronic hat, wie in letzter Zeit schon oft, die Vertriebsrechte erworben und bringt Starquake als Billigspiel unters Volk. Wer das Game noch nicht hat, der sollte es sich so schnell wie möglich besorgen. Gute Grafiken und flotte Musikuntermalung lassen so schnell keine Langeweile aufkommen. Starquake gibts für CPC, Commodore 64 und Spectrum 48K.



Super Trolley

Sie sind ein kleiner Hilfsangestellter in einem bekannten Supermarkt. Der Filialleiter kann Sie unglücklicherweise nicht ausstehen und schikaniert Sie, wo er nur kann. Heute hat er sich wieder was ganz Fieses ausgedacht: Erst müssen Sie diverse Waren sortieren, dann mit einem Einkaufswagen die Regale leeren und die Artikel anderswo wieder einräumen. Das alles natürlich in möglichst kurzer Zeit und ohne dabei die Kunden anzurempeln.



Bei Super Trolley, dem neuesten Billigspiel von Mastertronic, ist die Grafik zwar nicht gerade berauschend, aber noch ausreichend. Als kleine Entschädigung ertönt zu dem ganzen Spektakel eine flotte Hintergrundmusik, die vom Sound-Hexer Rob Hubbard programmiert wurde. Ob das Game auch Spaß macht, hängt vom persönlichen Geschmack ab. Computerneulingen kann man es vielleicht empfehlen, da die Aufgaben nicht allzu anspruchsvoll sind. Am besten, Sie sehen sich das Game vor dem Kauf genau an. Super Trolley gibt's für Commodore 64.

Subterranea – Rack it

In 16 Levels müssen Sie bei Subterranea Ihr Joystick-Gefühl unter Beweis stellen. Wieder einmal sind bösartige Aliens in unser friedliches Universum eingefallen. Die sollen Sie vertreiben, am besten durch kräftiges Drücken auf den Feuerknopf. Wie bei Airwolf steuern sie in diesem Game von Hewson ein Raumschiff durch die unterschiedlichsten Höhlensysteme. Dabei begegnen Ihnen ständig feindliche Flugobjekte, die sofort das Feuer eröffnen. Es empfiehlt sich, ebenfalls auf die Feuertaste zu drücken und zwar möglichst schnell, denn sonst sind Sie Ihr Raumschiff und damit auch eines der drei Leben unweigerlich los.

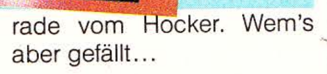
Subterranea könnte durchaus als Ballerspiel Erfolg haben, zumal auch Grafik und Sound nicht gerade schlecht sind. Leider jedoch existiert keine Möglichkeit, das Raumschiff mit dem Joystick zu steuern, so daß Sie sich mit einer äußerst umständlichen Tastaturbelegung die Finger verrenken dürfen. So kommt einfach nicht der rechte Spielspaß auf. Getestet haben wir Subterranea – Rack it auf dem Commodore 64 mit Farbmonitor.

Super Gran

Von Tynesoft, vor allem bekannt durch das Sportspiel

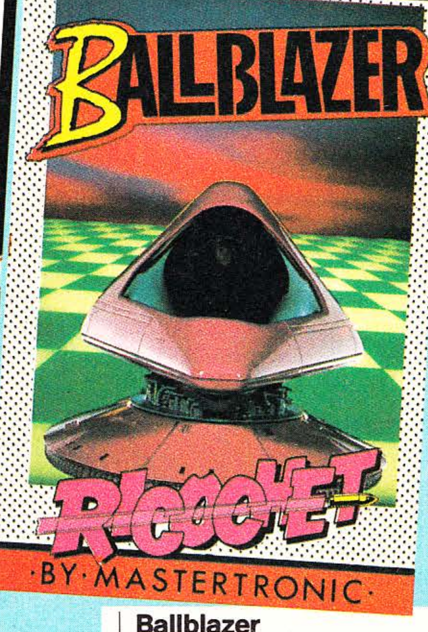
Winter Olympiad '88, kommt ein neues Spiel für den Commodore 16.
Bei Super Gran handelt es sich um das offizielle Computerspiel zur gleichnamigen englischen Fernsehserie. In einem amateurhaft gebastelten Helicopter müssen Sie einige Abenteuer bestehen. Grafisch schneidet Super Gran nicht sonderlich gut ab, da sich Sprites und Hintergrund kaum unterscheiden lassen. Auch das Spielprinzip reißt den Spieler nicht gerade vom Hocker. Wem's aber gefällt…





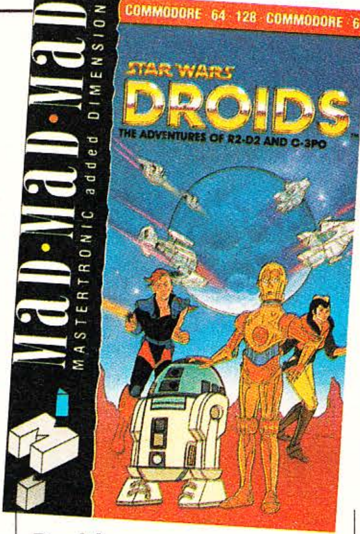
C16 Compilation

Eine preiswerte Spielesammlung bietet Mastertronic jetzt für alle C16/Plus4-Besitzer an. Auf der Kassette, die den eindrucksvollen Namen C16 Compilation trägt, sind vier Spiele, die in Sachen Grafik und Motivation eine preiswerte Alternative zu den meist überteuerten Spielesammlungen sind.
Im einzelnen handelt es sich um Pacmaina, ein Pac-Man-Verschnitt, der von der Geschwindigkeit her seinesgleichen sucht, Disasterblaster, ein gelungenes Weltraum-Ballerspiel, Laza, ebenfalls ein Ballerspiel, allerdings nicht ganz so motivierend, und schließlich das Sportspiel Downhill. Hier müssen Sie einen Abfahrtslauf hinter sich bringen. Für alle vier Spiele wird ein Joystick benötigt.



Ballblazer

Vor etwa 18 Monaten brachte die Firma Lucasfilm Games das Geschicklichkeitsspiel Ballblazer heraus, für das nun Mastertronic die Vertriebsrechte aufgekauft hat. Angeboten wird Ballblazer für den Commodore 64/128 und den Schneider PC. Es ist ein Spiel, bei dem der Bildschirm gesplittet ist, damit zwei Spieler gleichzeitig gegeneinander antreten können. Spaß macht Ballblazer allemal, so daß dem Kauf nichts im Wege steht. Beim CPC empfiehlt sich ein Farbmonitor.



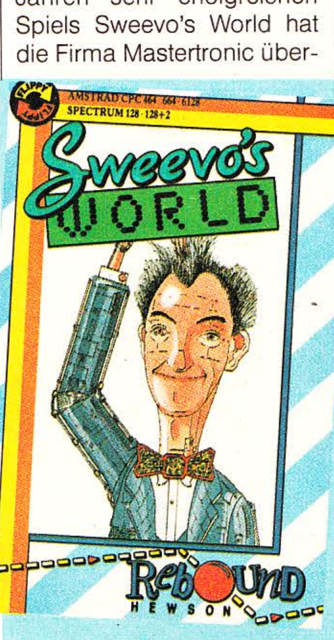
Droids

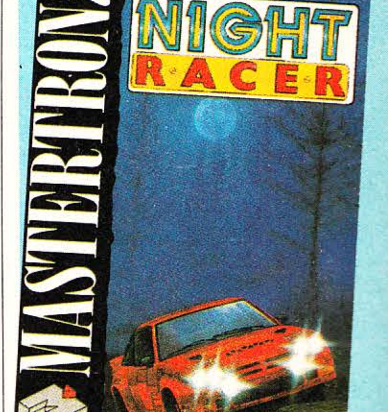
Droids ist ein neues Ballerspiel aus der Mastertronic-MAD-Serie, das durch hervorragende Grafiken und flotten Sound angenehm überrascht. Ihre Aufgabe ist es, die beiden Roboter R2-D2 und C-3P0 durch ein feindliches Raumschiff zu steuern (möchte bloß wissen, wie die da reingekommen sind!) und dabei so viele Aliens wie möglich zu zerstören. Wenn Ihnen die Namen der Robotis bekannt vorkommen: Ja, es sind die aus Star Wars, wie Sie am Untertitel erkennen können. Droids können Sie künftig auf dem Spectrum 48/128/Plus2 und dem Schneider CPC 464 spielen.

Sweevo's World

Den Vertrieb des vor zwei Jahren sehr erfolgreichen Spiels Sweevo's World hat die Firma Mastertronic übernommen und für den Schneider CPC und den Spectrum in neuer Verpackung herausgebracht. Sie übernehmen in diesem Game die Rolle der tollpatschigen Ente Sweevo, die ihren Planeten von bösartigen Sprites säubern muß. Sweevo's World ist ein typischer Vertreter der bekannten 3D-Adventures, wie es sie seit Knight Lore dutzendweise auf dem Markt gibt. Allerdings bietet das Spiel wesentlich bessere Grafiken als seine „Soft-Kollegen". Wer also noch nicht allzu viele Games dieser Kategorie zu Hause hat, darf sich Sweevo's World ohne Bedenken zulegen. Der Preis von knapp 15 Mark ist sicher nicht zu hoch.









Night Rallye

Eine Rallye-Simulation für den Commodore 64/128 bietet ebenfalls Mastertronic an. Allerdings darf man sich vom Titel nicht irreführen lassen, handelt es sich doch wieder einmal nur um ein simples Autorennspiel, das überhaupt nichts Neues bietet. Night Rallye kann nur allein und nur mit Joystick gespielt werden. Grafisch ist Night Rallye ganz gut gelungen, auch die Soundeffekte können sich hören lassen. Da das Spiel auch nur knapp über 10 Mark kostet, dürfte es eine interessante Alternative zu den teuren Autorennspielen sein. Thomas Bosch ■

Fortsetzung von S. 11

Spielfigur keine allzu große Anstrengung zumuten. Um ans Ziel zu kommen, ist es außerdem ratsam, jedes Objekt, sei es ein Stein, ein Baum oder eine Trinkwasserquelle, peinlich genau zu untersuchen.
Gesteuert wird die Spielfigur mit der Maus. Über die beiden Maustasten lassen sich auch durch Pulldown-Menüs die verschiedenen Aktionen anwählen. Ihre Spielfigur kann Objekte untersuchen, aufnehmen, ablegen, benutzen, verbinden und trennen. Außerdem können Sie sich jederzeit den Status Ihres Reporters betrachten. Dort finden Sie auch eine entsprechende Warnung, wenn der Held kurz vor der Erschöpfung steht.
Daß das abzusuchende Gebiet relativ groß ist, kann es ziemlich lange dauern, bis das Spielende erreicht ist. Glücklicherweise haben die Autoren auch Optionen zum Abspeichern und Wiedereinladen des aktuellen Spielstandes eingebaut. Hierzu muß sich eine frisch formatierte Diskette im Laufwerk A befinden.
Die Grafik von Bermuda-Projekt beschränkt sich zwar lediglich auf ein paar Objekte, ist aber völlig ausreichend. Da es notwendig ist, sämtliche Objekte zu untersuchen, würde sich das Spiel lange hinziehen, wenn der Bildschirm damit überfüllt wäre. Außerdem dürfen Sie nicht vergessen, daß Sie sich vorwiegend am Sandstrand einer einsamen Insel aufhalten, wo es keine Zivilisation gibt.
Zu Beginn des Spieles erklingt aus dem Lautsprecher ein digitalisiertes Musikstück. Ansonsten sind nur diverse Hintergrundgeräusche zu hören.

Fazit

Bermuda-Projekt ist ein Spiel für Leute mit viel Zeit und Geduld. Wer also einmal länger als eine Stunde an einem Computergame sitzen möchte und außerdem eine Vorliebe für Rätsel hat, dem kann ich dieses Spiel nur empfehlen. TB ■

Titel:	**Bermuda Project**
Getestet:	**Atari ST**
Umsetzungen:	**Amiga (in Vorbereitung)**
Im Test:	[X] Diskette, [ ] Kassette
Preis (DM):	59,–
[ ] **Joystick**	
[ ] **Tastatur**	
[X] **Maus**	

Wertung (Skala 0 – 25 – 50 – 75 – 100): Grafik, Sound, Bedienung, Motivation

Kurzberichte

Dieser Monat brachte leider nicht viele Neuerscheinungen und Umsetzungen für alle Computer. Sowohl für den C-64 als auch für Atari ST und Amiga hat wohl die Sommerpause eingesetzt.

Zwei Umsetzungen brachte Infogames für den Amiga auf den Markt. Beide sind schon seit einiger Zeit für andere Computer wie C-64 und ST erhältlich und wurden von uns getestet.

Chamonix Challenge

nennt sich ein Bergsteigerspiel, das zuerst für den kleinen Commodore erschien. Ziel des Spiels ist die Besteigung eines oder mehrerer Berggipfel. Dazu müssen sowohl Gletscher als auch Felswände erfolgreich überwunden werden. Das Game ist ein technisch hochwertiges und recht schwieriges Geschicklichkeitsspiel.
Die Grafik ist für Amiga-Maßstäbe jedoch enttäuschend ausgefallen. Sie unterscheidet sich nur geringfügig von der des Commodore 64. Die Schwierigkeit und der Spielwitz sind jedoch gleichgeblieben. Bergfans sollten sich Chamonix Challenge einmal ansehen. Für andere ist es empfehlenswert, erst einmal probezuspielen.
Das andere Game wurde vom Atari ST adaptiert:





Die Arche des Captain Blood

Captain Blood ist ein sehr kompliziertes Spiel, das in

keine Kategorie so recht einzuordnen ist. Aufgabe ist es, als Programmierer seine Einzelteile wieder aufzusammeln, die in einem Universum innerhalb des Computer verstreut sind. Dazu muß er Kontakt mit fremden Lebewesen aufnehmen. Exotische Planeten anzufliegen und gegebenenfalls auch zu zerstören ist hier angesagt.
Die Grafik ist identisch mit der des Atari ST. Das Spiel hat rasante Fraktalgrafik und herrliche Farbeffekte vorzuweisen. Musik und Sounduntermalung sind auch nicht zu verachten. Insgesamt eine gelungene Umsetzung eines gelungenen Spiels.



The Tree Stooges

Nur einen Kurztest war uns The Tree Stooges für den Amiga wert. Ziel des Spiels, das sich an der Fernsehserie orientiert, ist, für eine verarmte Familie Geld aufzutreiben. Dazu lasssen sich die drei Clowns allerlei einfallen.
Grafisch ist das Spiel allererste Güte, der Spielwitz bleibt jedoch auf der Null-Marke. Das, was sich die Programmierer haben einfallen lassen, mag ja als Grafikdemo ganz gut sein, für ein



Spiel ist jedoch nichts drin. Oder finden Sie daran Gefallen, wenn sich einige Leute minutenlange Tortenschlachten liefern oder ein Watschenspiel veranstalten, bei dem es einzige Aufgabe des Spielers bleibt, den Feuerknopf zu drücken?
Um ein gutes Spiel zu gestalten, hätten sich die Designer mehr einfallen lassen müssen als einige schwachsinnige Comicstrips mit teilweise recht langen Sequenzen dazwischen, in denen der Spieler nichts zu tun hat als kräftig zu gähnen.

Gauntlet II für Atari ST

Der glänzende Spielautomat und Nachfolger des Arcade-Games Gauntlet wurde jetzt auch für den Atari ST adaptiert. Die Umsetzung ist vom Automaten kaum zu unterscheiden. Das Game hat schnelle und hochauflösende Grafik und hervorragenden Sound vorzuweisen. Sogar an die Sprachausgabe des Automaten wurde gedacht. Ein besonderer Gag ist, daß auch am ST vier Spieler mitspielen können. Allerdings ist das leider nur mit einem speziellen Joystick-Interface möglich. Trotzdem ist diese Version wesentlich besser gelungen als sein Vorgänger Gauntlet I.

Atari-Umsetzung von ZARCH

Das bekannte Archimedes-Spiel Zarch können nun auch ST-Besitzer genießen. Das schnelle Aktiongame, bei dem außerirdische Wesen aller Art abgewehrt werden müssen, nennt sich auf dem ST Virus. Die Grafik bleibt im wesentlichen gleich, nur die Geschwindigkeit hat gelitten, was aber nichts heißen will. Das Game ist noch schnell genug, um einen fließenden Spielverlauf zu garantieren. Atari-Besitzer sollten sich Virus unbedingt ansehen.

Aktion-Game für Atari ST

She-Fox nennt sich ein neues Aktion-Spiel, das unter anderem auch für den ST erscheint. Das Game ist ein simples Scroll-Spiel, bei dem ausnahmsweise einmal eine Heldin die Hauptrolle spielt. Die Amazone muß sich am laufenden Band mit ihrer Peitsche gegen wilde Tiere und ein Zeitlimit behaupten. Grafisch ist das Spiel ganz gut gelungen, vom Spielwitz blieb allerdings nichts mehr übrig. Bereits nach einigen Spielminuten wird die ständige Raserei in eine vorgegebene Richtung langweilig. Insgesamt also ein sehr mittelmäßiges Spiel.

Pink Panther für den Commodore 64

Das in der letzten Ausgabe getestete Amiga-Spiel The Pink Panther wurde für den C-64 adaptiert. Kaum zu glauben: Die Grafik ist zwar nicht so detailreich, dafür aber schneller als auf dem Amiga. Auch der Sound ist für C-64-Maßstäbe hervorragend. Der Spielwitz hat nicht im geringsten gelitten. Aus dem kleinen Commodore haben die Programmierer herausgeholt, was er hergibt. Eine Umsetzung, wie sie nicht besser sein könnte. mn ■

Spiele

Players' Pages

Hallo, liebe Spielefreaks und Joystickartisten. Willkommen bei den Players Pages der LOAD & RUN. Wer Tips, Tricks, Pokes und Lösungen zu jedem x-beliebigen Spiel und für jedes x-beliebige System hat oder sucht, ist hier richtig!

Letzten Monat hat sich wieder viel getan bei uns. Entsprechend umfangreich fallen die Players Pages aus. Doch ich will mich nicht mit langen Vorreden aufhalten, sondern gleich loslegen.

Ranarama

Für Ranarama-Spieler auf dem C64/128 haben wir folgenden Poke bereit, der nach dem üblichen load-reset-poke-sys-Rezept eingebaut wird.

```
POKE 37104,96
POKE 33969,234
POKE 33970,234
```

Game Over Part II

Für Part II von Game Over finden viele kein passendes Codewort. Wie wär's denn mal mit ZAPPA? Übrigens werden für dieses Spiel noch jede Menge Strategie-Hinweise gesucht. Für alle Game-Over-Spieler auf dem Spectrum habe ich auch noch einen Code anzubieten: 18024.

Terramex

Haben Sie schon mal versucht, mit dem Ballon in die Lüfte zu steigen? Sicherlich, denn mit dem Schirm geht es ja einwandfrei. Aber wer kann uns verraten, wie der Ballon wieder zum Landen überredet werden kann? Die LOAD & RUN-Redaktion hat es auch nach mehreren durchwachten Nächten noch nicht geschafft. Übrigens: Wenn Sie es nicht schaffen, warum gehen sie dann nicht in den Untergrund. Wenn Sie in den Brunnen klettern, kommen noch jede Menge weitere Räume, in denen unzählige Gefahren lauern wie beispielsweise eine Schlange, die aber durch Flötenmusik beruhigt werden kann.

Die Erbschaft

Viele Leser haben sich über die Tips zum dritten Teil der Erbschaft in LOAD & RUN 4/88 gefreut. Andere dagegen konnten damit nichts anfangen, weil sie nicht mal den ersten Teil beenden konnten. Um ihrem Leiden nun ein Ende zu bereiten, hier eine Liste mit den Gegenständen, welche die einzelnen Personen erhalten:

Neger .... Trompete
Mann mit Glatze .... Taschenlampe
Mafioso .... Pistole
Chinese .... Kerzenhalter
Mann mit Scheiten .... Kaktus
Punkerin .... Bügeleisen
Mann mit Glatze & Bart .... Kette

Bei der Gelegenheit noch ein kleiner Tip: Es empfiehlt sich, stets zu Fuß zu gehen und den Aufzug zu meiden. Im Zimmer gleich links nach Ihrem eigenen Wohnraum finden Sie in der Schublade ein zusätzliches Taschengeld.

VERDIENEN SIE GELD MIT IHREM COMPUTER!

Haben Sie einen Commodore VC 20 oder C 64? Einen 16/116, Plus 4? Oder einen 128? Können Sie programmieren? In Basic oder Maschinensprache? Dann bietet COMMODORE-WELT Ihnen die Möglichkeit, mit diesem Hobby Geld zu verdienen!

Wie? Ganz einfach. Sie senden uns die Programme, die Sie für einen Abdruck als geeignet halten, zusammen mit einer Kurzbeschreibung, aus der auch die verwendete Hardware – eventuelle Erweiterungen – benutzte Peripherie – hervorgehen muß (Schauen Sie sich dazu den Kopf unserer Programmlistings an.)

Benötigt werden: Zwei Listings des Programms sowie eine Datenkassette oder Diskette! Wenn die Redaktion sich überzeugt hat, daß dieses Programm läuft und sich zum Abdruck eignet, zahlen wir Ihnen pro Programm je nach Umfang bis zu DM 300,–! Sollten Sie keinen Drucker haben, genügt der Datenträger.

Sie erhalten Ihre Kassette/Diskette selbstverständlich zurück, wenn Sie einen ausreichend frankierten Rückumschlag mit Ihrer Adresse beifügen.

Bei der Einsendung müssen Sie mit Ihrer Unterschrift garantieren, daß Sie der alleinige Inhaber der Urheber-Rechte sind! Benutzen Sie bitte anhängendes Formular! (Wir weisen darauf hin, daß auch die Redaktion amerikanische und englische Fachzeitschriften liest und „umgestaltete" Programme ziemlich schnell erkennt).

Um Ihnen die Arbeit zu erleichtern, finden Sie hier ein Formular. Sie können es ausschneiden oder fotokopieren.

Name des Einsenders: ______________________________

Straße/Hausnr./Tel.: ______________________________

Plz/Ort: ______________________________

Hiermit biete ich Ihnen zum Abdruck folgende(s) Programm(e) an:

____

____

____

____

Benötigte Geräte: ______________________________

____

Beigefügt ( ) Listings ( ) Kassette ( ) Diskette

Ich versichere, der alleinige Urheber des Programmes zu sein!
Hiermit ermächtige ich die Redaktion, dieses Programm abzudrucken und wirtschaftlich zu verwerten. Sollte es in den Kassetten-Service aufgenommen werden, erhalte ich auch dafür eine entsprechende Vergütung, das Copyright geht insoweit auf den Verlag über.

____

Rechtsverbindliche Unterschrift

COMMODORE WELT
PROGRAMM-REDAKTION
POSTFACH 1161
D-8044 UNTERSCHLEISSHEIM

- Linke Seitenfläche =
- Rechte Seitenflächen =
- Obere Seitenflächen =
- Untere Seitenflächen =

Nun steht der Programmierung nichts mehr im Wege. Tippen Sie das nachfolgende Listing analog zur Programmbeschreibung unter Einbeziehung der Variablenliste ab.

PROGRAMM-BESCHREIBUNG

In Zeile 100 bewirkt das GOSUB 9000 die Dimensionierung und Wertzuweisung für Variablen, die sich innerhalb des Programmablaufs nicht mehr ändern (Ausnahme: HS).
GOSUB 9100 schafft Ausgangswerte für eine neue Spielrunde.
GOSUB 8000 in Zeile 130 sorgt für die Spielfelddarstellung:
In Zeile 8000 werden drei Schleifen gebildet. Hierdurch können die sechsmal vier Einzelfelder mit fünf verschiedenen Farben belegt werden.
Die Farbwahl erfolgt durch Zufall in Zeile 8010.
Die Zeilen 8020 und 8030 bilden durch die Variable B eine zusätzliche Schleife, mit der es möglich ist, in Zeile 8020 festzustellen, ob der ermittelte Wert aus Zeile 8010 innerhalb des Einzelfeldes schon vorhanden ist. Ist dies der Fall, muß er neu gebildet werden: IF X = S%(B,H,V) THEN 8010.
Dieses Problem erklärt den langsamen Aufbau des Spielfeldes.
Wenn alle fünf Flächen eines Einzelfeldes fünf verschiedene Werte haben, wird im Unterprogramm 8200 (Zeile 8040) jeder Wert in eine augenfreundliche Farbe umgesetzt. Nun kann durch GOSUB 4000 das errechnete Einzelfeld dargestellt werden.
Die Zeilen 8060 und 8070 plazieren auf den unteren Bildschirmteil eine Kurzanleitung.
Die Zeile 8080 beschriftet die Punkte- und Highscore-Anzeigen; GOSUB 5100 und 5200 ergänzen dies mit den entsprechenden Werten.
Die Zeile 8100 sorgt in Verbindung mit Zeile 8110 für die farbliche Unterlegung der Farbtabelle.
In Zeile 200 wird festgestellt, wieviel Oberflächen von jeder Farbe vorhanden sind. Da die Farbtabelle mit den gleichen Variablen arbeitet, aus denen auch das Spielfeld besteht, muß der Wert der Oberfläche links oben mit der Variablen SR gespeichert werden, bevor innerhalb der Zeile 220 der Ausdruck der Farbwerte erfolgt (GOSUB 5000).
Danach wird S%(5,H,V) wieder auf den ursprünglichen Wert gebracht: S%(5,H,V) = SR.
Die nachfolgende Joystickabfrage (siehe auch Handbuch Seite 185) erfolgt innerhalb der Zeilen 300 bis 330. Zweckmäßigerweise erfolgt die Zugwahl unter der Bedingung, die Ränder des Spielfeldes nicht zu überschreiten, zum Beispiel:
IF JOX(1) = 1 (also nach oben) AND V 1 in Zeile 300. Falls dies nicht zutrifft, bewirkt GOSUB 2100 die Löschung des Cursors an alter Position. Danach erfolgt eine Verzweigung in richtungszugehörige Unterprogramme (1000, 1100, 1200 und 1300).

SUBROUTINEN

Anhand des Unterprogramms 1000 soll die Farbverschiebung nach oben als Beispiel beschrieben werden:
Sowohl die grafische vertikale Spielfeldposition (VG) als auch der rechnerische Wert (V) wird um eine Position nach oben hin reduziert. Die drei Farben, die verschoben werden sollen, werden durch die Variablen S1, S2 und S3 gespeichert.
GOSUB 1400 verringert den Farbwert der bisherigen Oberfläche auf der Tabelle um eins.
Mit Hilfe der Speichervariablen werden die Farbwerte in Zeile 1010 neu verteilt, so daß eine Verschiebung stattfindet.
GOSUB 4000 stellt danach das neu errechnete Einzelfeld dar.
GOSUB 1500 aktualisiert die Farbtabelle und erhöht die Punktzahl.

In unserem Beispielprogramm wird viel mit dimensionierten Variablen gearbeitet. Gerade am Anfang ist es schwer, einzusehen, warum man Variablen überhaupt dimensionieren soll. Leider gibt das Handbuch auf den Seiten 147 und 148 nur wenig Auskunft auf diese Frage. Hier die wichtigsten Gründe:

1. Dimensionieren ist praktisch.
Stellen Sie sich vor, Sie möchten 20 Variablen mit Werten belegen. Das sähe etwa so aus:

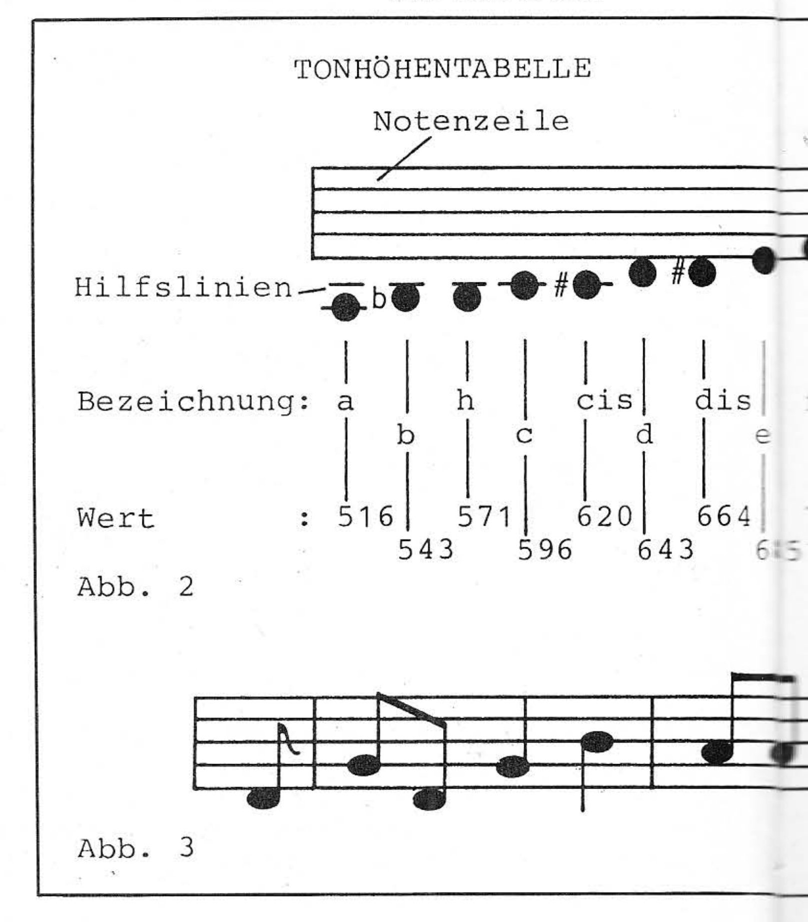


Abb. 2

Abb. 3

Nach der Rückkehr wird am Ende der Joystickabfrage in Zeile 350 der Cursor durch GOSUB 2000 an die neue Position gesetzt. Falls der Zählwert der aktuellen Oberflächenfarbe auf 24 steht, ist die Aufgabe erfüllt und das Programm verzweigt zur End-Routine (Zeile 9500 bis 9550), andernfalls zum Anfang der Joystickabfrage (GOTO 300).

A1=2: A2=4: A3=7: A4=9 und so weiter.
Durch eine dimensionierte Variable läßt sich das Problem viel eleganter lösen:
DIM A(20): FOR B = 1 TO 20 : READ(A): NEXT DATA 2, 4, 7, 9 und so weiter.

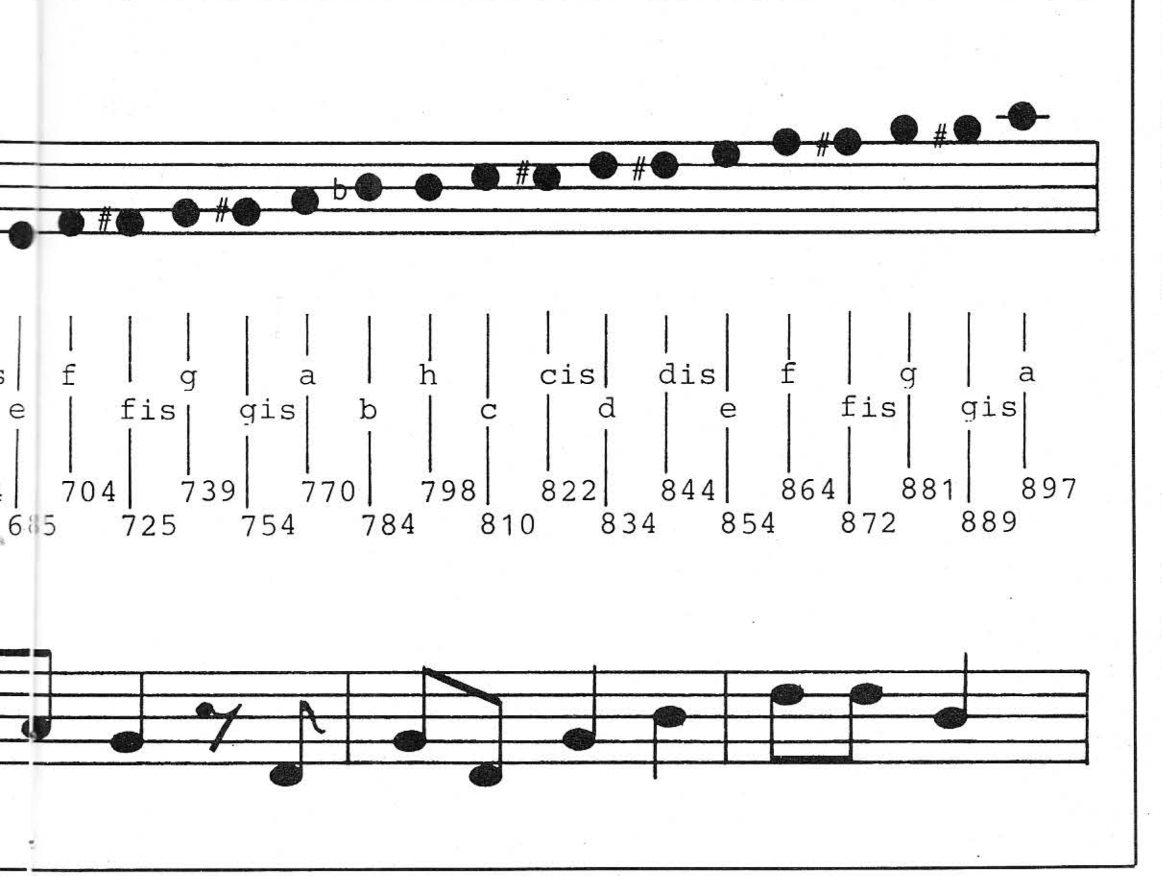
2. Dimensionieren eröffnet neue Möglichkeiten.
Bei dem Beispiel DIM A (20) reservierten Sie sich

einen Speicherplatz für 20 Variablen gleichen Namens. Dies geschah in eindimensionaler Form, womit Sie praktisch eine Variablenreihe geschaffen haben (siehe PAINTER Zeile 220 und 230). Sie können aber auch ein Variablenfeld schaffen: DIM A(20,20). Sie haben sich dadurch 20 mal 20, also 400 Variablen zweidimensional reserviert. Natürlich können Sie auch drei-, vier- oder fünffach dimensionieren – bei PAINTER waren es sechs mal vier Einzelfelder mit fünf Farben, also S%(5,6,4) –, wenn der Speicherplatz dies mitmacht.

Diese Verfahrensweise erlaubt es unter anderem, mit Hilfe anderer Variablen innerhalb Ihres dimensionierten Feldes an jede beliebige Position zu gelangen und dadurch komplizierte Spielabläufe oder Sortierungen zu gestalten. Bei PAINTER waren es die Variablen H und V innerhalb des Zahlenfeldes S%.

3. Dimensionieren spart Speicherplatz.
Abgesehen von der Programmverkürzung, die aus dem vorhergehenden Argument resultiert, ist es speicherplatzsparend, wenn Ihr Computer nur eine Variable mit Reservation einrichten muß anstatt eine entsprechende Variablen-Anzahl mit verschiedenen Namen.
Sie sehen also, daß man in vielen Fällen auf die Dimensionierung kaum verzichten kann. Bei geringer Anzahl von Variablen kann es übersichtlicher und praktischer sein, die Dimensionierung zu lassen, siehe PAINTER Zeile 1000 und 1010.
Natürlich sind auch Strings dimensionierbar und lassen sich dadurch ähnlich wie Variablen handhaben, etwa bei Sortiervorgängen.
Falls es sich bei dimensionierten Variablen in jedem Fall um ganze Zahlen handelt, sollten Sie dies durch % markieren; daher S%(5,H,V) statt S(5,H,V). Sie sparen dadurch weiteren Speicherplatz und erhöhen die Rechengeschwindigkeit.
Beachten Sie, daß Sie innerhalb eines Programmablaufs eine Variable nur einmal dimensionieren können, andernfalls erfolgt die Fehlermeldung REDIM 'D ARRAY ERROR. Um alle Variablen eines Feldes wieder auf null zu bringen, ist demnach eine entsprechende Schleife nötig (siehe PAINTER Zeile 9110). Dieser Punkt entfällt beim Programmstart durch RUN oder nach einem CLEAR-Befehl.
Wenn Sie das Beispielprogramm ausprobieren, werden Sie zur Lösung der Aufgabe immer weniger Züge brauchen.
Auf diese Weise werden immer mehr strategische Überlegungen zum Tragen kommen. Nach einiger Zeit beherrschen Sie diverse Routinezüge, wodurch das Spiel kaum noch eine Anforderung darstellt. Es liegt in Ihrer Hand, diese Anforderung durch Änderungen der Spielaufgabe wesentlich zu erhöhen.
Es wird viel schwieriger, wenn Sie das Programm so ändern, daß Züge in unmittelbarer Gegenrichtung, etwa links–rechts, nicht erlaubt sind. Hierzu könnten Sie der Jostickabfrage durch AND eine weitere Bedingung außer der Spielfeldgebundenheit anfügen. Auch andere Änderungen der Spielregeln wären denkbar; es steht noch genug Speicherplatz zur Verfügung, zum Beispiel ab Zeile 3000.
Auch Titel und Melodien fehlen noch und warten auf Ihre Programmierkünste.
Die programmtechnischen Voraussetzungen für die Sound-Programmierung sollten Ihnen bekannt sein und lassen sich notfalls im Handbuch schnell nachlesen (Seite 109 bis 116).
Wie Sie wissen, verlangt der SOUND-Befehl die Festlegung von Tonkanal (1 bis 3), Tonhöhe (0 bis 1015) und Tondauer (1 bis ?).
Von den drei Tonkanälen ist nur der erste und zweite Kanal für Melodien innerhalb der BASIC-Programmierung brauchbar. Glücklicherweise erlauben es die Hardware-Voraussetzungen des C16, daß während der Tondauer andere Programmschritte abgearbeitet werden können. Insofern ist es möglich, auf Kanal zwei die Begleitstimme zu Kanal eins erklingen zu lassen und damit zwei Töne zur gleichen Zeit zu erzeugen.

Die Tonhöhe wird im Notensystem durch die vertikale Position einer Note innerhalb der Notenzeile (fünf Linien) festgelegt. Je tiefer sich die Note innerhalb der Zeile befindet, um so tiefer klingt sie auch (und umgekehrt). Wenn's ganz



PAINTER-STRATEGIESPIEL

```
10 rem painter=================c16 <io>
11 rem (p) commodore welt team      <mj>
12 rem ============================ <pn>
13 rem (c) by peter bergen          <fc>
14 rem         hildesheim           <la>
15 rem                              <kn>
16 rem beispielprogramm 2 zur       <lp>
17 rem serie "so programmiere ich   <ap>
18 rem          in basic"           <km>
19 rem                              <lf>
20 rem basic v3,5                   <in>
21 rem c16/116/p4                   <lj>
22 rem ============================ <ih>
95 gosub 60020                      <kd>
100 gosub9000                       <go>
110 gosub9100                       <kn>
120 vol6:color0,1:color4,1:scnclr   <jm>
130 gosub8000                       <dk>
199 rem wertzuweisungen             <mj>
200 forv=1to4:forh=1to6:f%(s%(5,h,
v))=f%(s%(5,h,v))+1:next:next       <ne>
210 h=1:v=1:hg=1:vg=1:sr=s%(5,h,v)  <fc>
220 restore220:fora=1to5:readb:s%(
5,h,v)=b:gosub5000:next:s%(5,h,v)=
sr                                  <pp>
230 data2,3,6,7,10                  <pe>
299 rem joystickabfrage             <ac>
300 ifjoy(1)=1andv>1thengosub2100:
gosub1000:goto350                   <dk>
310 ifjoy(1)=3andh<6thengosub2100:
gosub1100:goto350                   <oo>
320 ifjoy(1)=5andv<4thengosub2100:
gosub1200:goto350                   <da>
330 ifjoy(1)=7andh>1thengosub2100:
gosub1300:goto350                   <ja>
350 gosub2000:iff%(s%(5,h,v))=24th
en9500:elsegoto300                  <hb>
999 rem verschiebung nach oben      <cj>
1000 vg=vg-5:v=v-1:s1=s%(1,h,v):s2
=s%(5,h,v):s3=s%(3,h,v):gosub1400   <mg>
1010 s%(1,h,v)=s2:s%(5,h,v)=s3:s%(
3,h,v)=s1:gosub4000:gosub1500:retu
rn                                  <df>
1099 rem verschiebung nach rechts   <cb>
1100 hg=hg+5:h=h+1:s1=s%(4,h,v):s2
=s%(5,h,v):s3=s%(2,h,v):gosub1400   <ha>
1110 s%(4,h,v)=s3:s%(5,h,v)=s1:s%(
2,h,v)=s2:gosub4000:gosub1500:retu
rn                                  <ja>
1199 rem verschiebung nach unten    <jm>
1200 vg=vg+5:v=v+1:s1=s%(1,h,v):s2
=s%(5,h,v):s3=s%(3,h,v):gosub1400   <cc>
1210 s%(1,h,v)=s3:s%(5,h,v)=s1:s%(
3,h,v)=s2:gosub4000:gosub1500:retu
rn                                  <pa>
1299 rem verschiebung nach links    <fl>
1300 hg=hg-5:h=h-1:s1=s%(4,h,v):s2
=s%(5,h,v):s3=s%(2,h,v):gosub1400   <gc>
1310 s%(4,h,v)=s2:s%(5,h,v)=s3:s%(
2,h,v)=s1:gosub4000:gosub1500:retu
rn                                  <bl>
1399 rem farbtabellen-abzug         <be>
1400 f%(s%(5,h,v))=f%(s%(5,h,v))-1
:gosub5000:return                   <el>
1499 rem aktualisierung             <ef>
1500 f%(s%(5,h,v))=f%(s%(5,h,v))+1
:gosub5000:p=p+1:gosub5100:return   <bb>
1999 rem kursor setzen              <lg>
2000 color1,2:char1,hg+2,vg+2,q2$+
"*":return                          <bj>
2099 rem kursor loeschen            <gi>
2100 color1,s%(5,h,v),4:char1,hg+2
,vg+2,q1$+" ":return                <jd>
3999 rem felddarstellung            <ol>
4000 color1,s%(1,h,v),3:char1,hg+1
,vg,q1$+z2$+z2$+z2$                 <nj>
4010 fora=vg+1tovg+3                <oe>
4020 color1,s%(4,h,v),3:char1,hg,a
,z4$:color1,s%(5,h,v),4:char1,hg+1
,a,"   "                            <ic>
4030 color1,s%(2,h,v),3:char1,hg+4
,a,z6$                              <id>
4040 next:color1,s%(3,h,v),3:char1
,hg+1,vg+4,z3$+z3$+z3$:return       <mf>
4999 rem farbkontrolle              <gf>
5000 color1,s%(5,h,v),4:char1,34,f
p%(s%(5,h,v)),q1$:printusing"##";f
%(s%(5,h,v))                        <ec>
5010 return                         <nj>
5099 rem punktanzeige               <gh>
5100 color1,8,6:char1,32,3,q1$:pri
ntusing"######";p:sound1,100,2:ret
urn                                 <ci>
5199 rem highscoreanzeige           <mk>
5200 color1,8,6:char1,32,6,q1$:pri
ntusing"######";hs:return           <cl>
7999 rem spielfelddarstellung       <cp>
8000 forv=1to4:forh=1to6:fora=1to5  <fo>
8010 x=int(rnd(1)*5)+1:b=1          <in>
8020 ifx=s%(b,h,v)then8010          <hc>
8030 b=b+1:ifb=6then8040:else8020   <ea>
8040 s%(a,h,v)=x:next:gosub8200:go
sub4000:hg=hg+5:next:hg=1:vg=vg+5:
next                                <ka>
8060 color1,2:char1,1,22,q2$+"sie
haben gewonnen, wenn alle quadrate
"                                   <ha>
8070 char1,1,23,"gleichfarbig sind
! steuerung mit joy(1)"             <dp>
8080 color1,8,4:char1,32,2,q1$+"pu
nkte":char1,32,5,q1$+" high "       <hd>
8090 gosub5100:gosub5200            <lk>
8100 restore8110:fora=1to5:readb:c
olor1,b,4:char1,33,fp%(b),q1$+"
```

```
":next:return                          <ag>
8110 data2,3,6,7,10                    <co>
8199 farbwertzuweisung                 <dg>
8200 forb=1to5                         <ck>
8210 ifs%(b,h,v)=1thens%(b,h,v)=2:
goto8260                               <mk>
8220 ifs%(b,h,v)=2thens%(b,h,v)=3:
goto8260                               <lk>
8230 ifs%(b,h,v)=3thens%(b,h,v)=6:
goto8260                               <pi>
8240 ifs%(b,h,v)=4thens%(b,h,v)=7:
goto8260                               <no>
8250 ifs%(b,h,v)=5thens%(b,h,v)=10    <jn>
8260 next:return                       <cl>
8999 rem dimensionierung und wertz
uweisungen                             <nd>
9000 a=rnd(1)-ti:dims%(5,6,4):dimf
%(10):dimfp%(10):hs=1000               <ac>
9010 fp%(2)=10:fp%(3)=12:fp%(6)=14
:fp%(7)=16:fp%(10)=18                  <ge>
9020 q1$=chr$(18):q2$=chr$(146):re
turn                                   <gk>
9099 rem ausgangswerte                 <pi>
9100 fora=1to10:f%(a)=0:next           <ji>
9110 forv=1to4:forh=1to6:fora=1to5
:s%(a,h,v)=0:next:next:next            <fb>
9120 p=0:hg=1:vg=1:return              <mn>
9499 rem ende                          <bn>
9500 color1,2:char1,1,22,q2$+"
sie haben es geschafft!
"                                      <pf>
9510 char1,1,23,"noch ein spiel? (
li=ja / re=nein)        "              <dh>
9520 ifp<hsthenhs=p:gosub5200          <og>
9530 ifjoy(1)=7thenscnclr:goto110      <lf>
9540 ifjoy(1)=3thenscnclr:end          <fa>
9550 goto9530                          <cd>
60000 rem nachspann ==============     <pj>
60010 rem ** zeichensatz/graphik *     <ib>
60020 z2$=chr$(163):z3$=chr$(164)      <oo>
60030 z4$=chr$(165):z6$=chr$(167)      <ig>
60040 return                           <hc>
60050 rem ======================       <ei>
60060 rem 12277 bytes memory           <pe>
60070 rem 03544 bytes program          <jd>
60080 rem 00112 bytes variables        <mo>
60090 rem 00489 bytes arrays           <bi>
60100 rem 00333 bytes strings          <oh>
60110 rem 07799 bytes free (0)         <lb>
60120 rem ======================       <id>
```

extrem wird, stehen nach oben und nach unten noch Hilfslinien zur Verfügung. Aus kommunikativen Gründen tragen die Noten Buchstaben, damit sich erklären läßt, welche Tonhöhe gemeint ist.
Für weitere Differenzierungen stehen Vorzeichen zur Verfügung, die, vor die betreffende Note gesetzt, diese um einen Halbton vermindern oder erhöhen. Ein kleines „b" bedeutet eine Verminderung, ein Kreuz (#) eine Erhöhung. Mit dem Vorzeichen ändert sich auch die Bezeichnung der Note. Das erhöhte „f" heißt „fis", das verminderte „h" heißt „b". Die Abbildung zeigt Ihnen die gebräuchlichsten Tonhöhen zusammen mit ihren Bezeichnungen und den dazugehörigen Werten für den SOUND-Befehl. Stehen Vorzeichen schon am Anfang der Notenzeilen, so gelten sie für alle betreffenden Noten.
Bei einem Kreuz am Zeilenanfang an der Position der Note „f" wird so aus jedem „f" ein „fis". Soll im Einzelfall eine durch Vorzeichen am Zeilenanfang erhöhte oder verminderte Note dennoch normal gespielt werden, so wird durch das Zeichen ♮ die Erhöhung oder Verminderung für die folgende Note aufgehoben.
Die Tondauer wird dadurch bestimmt, welches Notenzeichen verwendet wurde. Die Abbildung 1 zeigt die Noten- und Pausenzeichen mit ihren Bezeichnungen und welche Werte der SOUND-Befehl für das jeweilige Zeichen benötigt.
Grundsätzlich ist es egal, ob die sogenannten Notenhälse nach oben oder nach unten zeigen; entscheidend ist, in welcher Höhe sich der „Kopf" befindet.
Auch bei der Tondauer gibt es eine Besonderheit: Befindet sich hinter der Note ein Punkt, so wird sie um die Hälfte ihrer Dauer verlängert.
Balken, Bögen, Notenschlüssel, Taktangaben (Zahlen am Anfang einer Notenzeile) und andere Zeichen dürfen Sie für unseren Zweck getrost vernachlässigen.

Zum Programmtechnischen:
Hier bietet sich die DATA-Methode an. Bleibt nur noch die Frage, wie man die Pausen programmiert. Aus dem Handbuch geht hervor, daß Tonhöhen innerhalb der Werte 0 bis 1015 liegen sollen. Geben Sie VOL 6: SOUND 1, 1022, 50 ein. Auch das wird ohne Fehlermeldung geschluckt. Nur hört man nichts, insofern die ideale Pause. Es hindert also nichts mehr daran, die ersten Versuche zu wagen. Probieren Sie, die Notenzeile der Abbildung 3 in ein kleines Musikprogramm zu verwandeln. Sollten zwei Noten gleicher Höhe aufeinanderfolgen, genügt ein SOUND 1, 1022, 1, um sie getrennt erklingen zu lassen.
Wenn das „schöne Frühjahr" auch bei Ihnen erklungen ist, sollte Ihr Programm etwa so aussehen:

```
10 VOL 6: FOR A = 1
   TO 19: READ B,C :
   SOUND 1,B,C: NEXT
20 DATA 643,8,739,15,
   643,15,739,15,798,
   15,770,15,1022,1,770,
   15,739,15
30 DATA 1022,8,643,8,
   739,15,643,15,739,15,
   798,15,834,15,1022,
   1,834,15,798,15
```

Falls Sie die langsame Gangart bevorzugt haben, sind Sie zu anderen Tonlängen gekommen.
Das Ganze klingt wesentlich voller, wenn Sie hinter SOUND 1,B,C noch ein SOUND 2,B−1,C setzen.

Auch die Programmierung der zweiten Stimme ist keine schwere Übung. Die betreffenden Noten stehen innerhalb der Notenzeile unter der ersten Stimme, meist mit den Hälsen nach unten. Sie müssen nur noch dafür sorgen, daß innerhalb einer Schleife nicht zwei, sondern vier DATAs (für SOUND 2) eingelesen werden.
Mehr brauchen Sie nicht, um Ihre Programmsammlung melodiös zu bereichern.

Peter Bergen □

Die Schildkröte in deutsch

Ein ganz besonderer Leckerbissen ist das Programm Igel-Grafik, eine Programmiersprache zur Erstellung von Grafiken.

Der Igel ist ein kybernetisches Tier, das auf dem Bildschirm lebt und Befehle ausführt, die ihn zum Laufen (vorwärts oder rückwärts) oder zum Drehen um die eigene Achse (rechts oder links) veranlassen. Bei der Bewegung zeichnet der Igel seine Spur auf den Bildschirm. Er kann also dazu benutzt werden, Zeichnungen darauf zu erzeugen.
Igelprogramme werden gebildet durch Zusammenfassen von Grundbefehlen zu sogenannten Prozeduren; diese Programme können wieder in anderen Prozeduren verwendet werden, so daß schrittweise bis zu einem beliebigen Grad an Komplexität aufgebaut werden kann. Jeder einzelne Schritt kann aus einem Grundbefehl oder aus einer vom Benutzer entwickelten Prozedur bestehen.
Alle Prozeduren, ob als Grundbefehl vorhanden oder vom Benutzer definiert, können ausgeführt werden, indem ihre Namen einfach mit der Tastatur eingegeben werden. Der im Programm integrierte Editor macht es leicht, Prozeduren zu definieren, zu verändern und ausfzuführen.

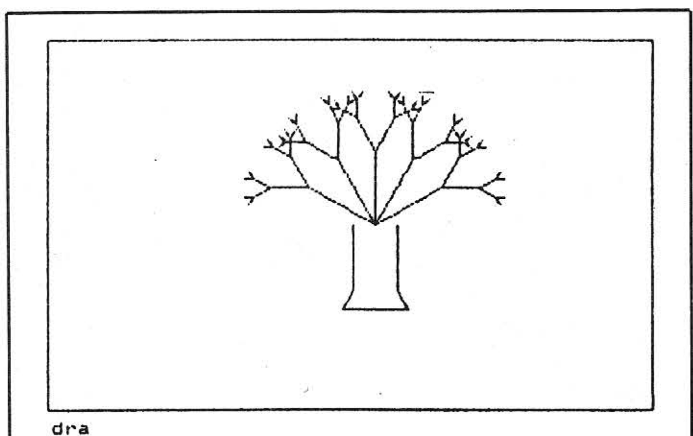
PROGRAMMSTART

Etwas Geduld ist nach dem Programmstart erforderlich, bis die Datazeilen der Maschinenroutine eingelesen sind. Alle für das Hauptprogramm später nicht mehr benötigten BASIC-Zeilen werden durch eine Maschinenroutine gelöscht. Das schafft Platz im Speicher.

HARDWARE

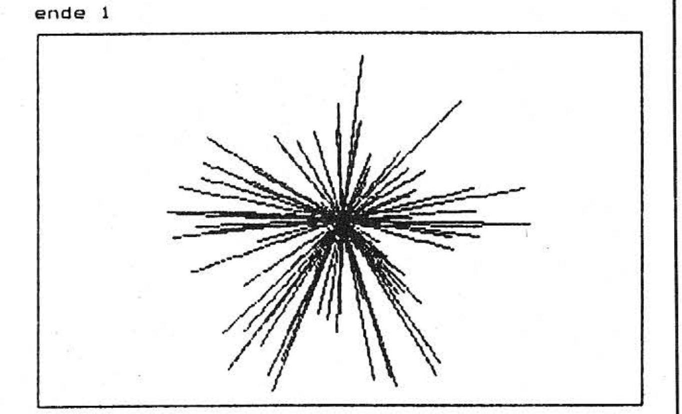
- Computer Plus/4 oder C16/116 mit 64 K-Byte.
- Floppy.
- Drucker Epson-kompatibel für den Ausdruck von Grafiken; für den Ausdruck von Prozeduren reicht auch ein anderer Drucker. Durch Benützung einer geeigneten Hardcopy-Routine können die Grafiken aber auch mit jedem sonstigen Drucker zu Papier gebracht werden.

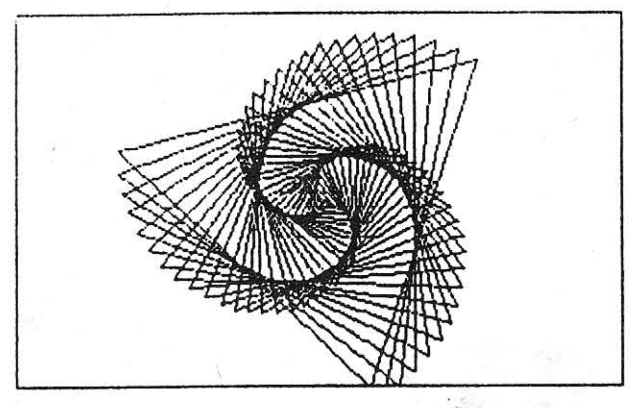
Auch wer nur eine Datasette besitzt, kann mit der Igel-Grafik experimentieren. Allerdings sind die Möglichkeiten sehr beschränkt, wenn auf das Laden und Speichern von Prozeduren und Bildern verzichtet werden muß.



```
dra

pr zufall
var.laenge+
var.winkel+
wh 100
zz-l 100
vw
rw
zz-w 360
re
endwh
ende 1
```





```
wiederhole 100     laenge+ 1
vi                 poly
winkel= 88         db
```

BEFEHLSLISTE IM DIREKTMODUS

Funktionstasten

F1 voller Grafikbildschirm;
F2 neun Zeilen Text;
F3 fünf Zeilen Text;
F5 Neustart Igelprogramm;
F8 voller Textbildschirm.
Die Funktionstasten F1, F2, F3 und F8 sind interruptprogrammiert. Daher ist ihre Bedienung jederzeit im laufenden Programm möglich.

Grundbefehle

ade
Alle in das Programm aufgenommenen Prozeduren werden gelöscht. Hierzu muß das Programm DELETE auf der Diskette vorhanden sein.

basic
Prozeduren werden im BASIC-Editor gezeigt. Zu sehen ist nicht die eingegebene Form, sondern wie diese in BASIC übersetzt wurde. Editieren ist möglich.

zeige.titel (zt)
Alle Namen der aufgenommenen Prozeduren werden gelistet.

versteckigel (vi)
Der als Dreieck auf dem

Bildschirm sichtbare Igel wird unsichtbar. Programme können dadurch schneller abgearbeitet werden.

zeigigel
Der unsichtbar gemachte Igel wird wieder sichtbar.

igeltext [1]
Der danach eingegebene Text wird auf den Grafikbildschirm ausgegeben. Folgt dem Befehl igeltext der Parameter 1, so wird der auszugebende Text revers dargestellt.

wiederhole (wh) <Anzahl>
Durch diesen Befehl, dem eine Zahl nachfolgen muß, wird die Anzahl der Wiederholungen bestimmt.

fuelle
Eine geschlossene Fläche, innerhalb derer sich der Igel befindet, wird mit der Zeichenfarbe gefüllt.

A. Befehlsliste

warte <zahl>	Programmverzögerung
stifthoch (sh)	
stiftab (sa)	
zeigigel (zi)	wie Direktmodus
versteckigel (vi)	wie Direktmodus
zufallszahl–1 (zz-l) <Bereich>	zufällige Igelschrittlänge erzeugen
zufallszahl–w (zz-w) <Bereich>	zufälligen Igeldrehwinkel erzeugen
aufkurs (ak)	wie Direktmodus
igeltext	wie Direktmodus
fuelle	wie Direktmodus
ls	Textschirm löschen, Prozedureneingabe
vorwaerts (vw) <Länge>	wie Direktmodus
rueckwaerts (rw) <Länge>	wie Direktmodus
links (li) <Winkel>	wie Direktmodus
rechts (re) <Winkel>	wie Direktmodus
stop.wenn.winkel> (sww>)	Abbruchbedingung
stop.wenn.winkel< (sww<)	Abbruchbedingung
stop.wenn.winkel= (sww=)	Abbruchbedingung
stop.wenn.laenge> (swl>)	Abbruchbedingung
stop.wenn.laenge< (swl<)	Abbruchbedingung
stop.wenn.laenge= (swl=)	Abbruchbedingung
var.laenge+	Länge Igelschritt variabel wählen
var.laenge*	Länge Igelschritt variabel wählen
var.winkel+	Drehwinkel Igel variabel wählen
var.winkel*	Drehwinkel Igel variabel wählen
winkel+	Winkelzuwachs wählen
winkel*	Winkelzuwachs wählen
laenge+	Längenzuwachs wählen
laenge*	Längenzuwachs wählen
wiederhole <Anzahl>	Anzahl der Wiederholungen bestimmen. Ist keine Anzahl angegeben, so wird diese vor dem Prozeduraufruf bestimmt.
ende.wiederhole	Schleifenabschluß
mitte	Igel geht zur Mitte und hinterläßt eine Spur
loeschebild	wie Direktmodus
pause	Programmunterbrechung. Fortsetzung auf Tastendruck
laenge= <Zahl>	Schrittlänge setzen
winkel= <Winkel>	Drehwinkel setzen
aufx <Koordinate>	Igel auf X-Position setzen
aufy <Koordinate>	Igel auf Y-Position setzen
radiere	Igel löscht Spur, die er überfährt
farbe <Nummer>	Spurfarbe wählen
luminanz <Zahl>	wie Direktmodus
rahmen <Zahl>	wie Direktmodus
ende <Zahl>	Prozedurennummer wählen

inhalt (ih)
Das Disketten-Directory wird aufgelistet.

vorwaertssh (vwsh) <Länge>
Vorwärts ohne Spur.

rueckwaertssh (rwsh) <Länge>
Rückwärts ohne Spur.

ls
Textbildschirm wird gelöscht.

rueckwaerts (rw) <Länge>
(selbsterklärend)

vorwaerts (vw) <Länge>
(selbsterklärend)

rechts (re) <Winkel>
(selbsterklärend)

links (li) <Winkel>
(selbsterklärend)

invert (i)
Grafikbildschirm wird invertiert.

xko
Ausgabe der X-Koordinate der Igelposition.

yko
Ausgabe der Y-Koordinate der Igelposition.

schiebbild <Schirmnummer>
Verschieben eines Bildes nach Schirm zwei oder drei (insgesamt drei Grafikbildschirme).

holbild <Schirmnummer>
Holen eines Bildes von Schirm zwei oder drei (Weiterbearbeitung möglich).

zeigbild <Schirmnummer>
Bild zeigen auf Schirm eins, zwei oder drei.

winkel= <Größe>
Winkelgröße setzen.

winkel+ <Zuwachs>
Zuwachs für den Winkel setzen.

laenge+ <Zuwachs>
Zuwachs für die Länge setzen.

radiere
Pixel löschen.

luminanz (lu) <Größe>
Farbhelligkeit setzen.

farbe <Code>
Zeichenfarbe wählen.

rahmen <Code>
Rahmenfarbe wählen.

aufxy
X- und Y-Koordinaten für Igelposition setzen. Der Rechner fragt anschließend nach den Koordinatenwerten. Es ändert sich nur die Position, nicht aber der Kurs des Igels.

kurs
Ausgabe des Igelkurses (Winkel 0 bis 360).

aufkurs (ak) <Winkel>
Igelkurs setzen.

vergiss (vg) <Prozedurnummer>
Prozedur aus dem Arbeitsspeicher löschen.

loeschefile <Name>
File auf Diskette löschen.

druckebild (db)
Grafik drucken von Schirm eins. Nur mit Epson-kompatiblem Drucker möglich.

druckerein (dre)
Protokoll ein. Mit jedem Drucker möglich.

druckeraus (dra)
Protokoll aus.

ladebild [1] <Name>
Grafik von Diskette laden. Folgt dem Befehl der Parameter 1, so werden nur 33 Blocks für die reine Bildinformation geladen. Wird der Parameter 1 weggelassen, so werden 41 Blocks geladen, die auch die Farbinformation enthalten.

bewahrebild [1] <Name>
Grafik auf Diskette speichern. (33 Blocks bei Parameter 1, sonst 41 Blocks.)

lade <name>
Prozeduren laden (Paket). Die Prozeduren müssen bereits in BASIC übersetzt vorliegen.

bewahre <name>
Prozeduren auf Diskette sichern. Alle im Arbeitsspeicher vorliegenden Prozeduren werden als Prozedurenpaket gespeichert und können zu einem späteren Zeitpunkt mit dem Lade-Befehl wieder eingeladen werden.

loeschebild (lb)
Grafikbildschirm löschen. Die Igelposition bleibt erhalten.

mitte
Igel in Bildschirmmitte setzen (ohne Spur).

bild
Grafikbildschirm löschen (Igel in Bildschirmmitte).

edit
Übergang in den Prozeduren-Editor.

ende
Programmabbruch.

1. Im Direktmodus edit eingeben

2. Prozedurnamen wählen <RETURN>
<ESC>-Taste bei Redigierwunsch bereits erstellter und auf Diskette gespeicherter Prozeduren, sonst <RETURN>. Prozeduren werden als sequentielle Files gespeichert und müssen in den Prozedurenspeicher, wie weiter unten beschrieben, eingelesen werden. Im Unterschied dazu werden Prozedurenpakete, die aus einer Vielzahl fertig eingespeicherter Prozeduren bestehen, durch den Befehl „lade“ aus dem Direktmodus heraus geladen und mit „bewahre“ gesichert.

3. Befehle eingeben
- Ein Befehl pro Zeile.
- Befehl und Parameterzahl immer durch ein Leerzeichen trennen, sonst erfolgt Fehlermeldung.
- Maximal neun Zeilen sind möglich, sonst erfolgt Fehlermeldung.
- Beliebiges Umherwandern und Verändern der Eingabe am Bildschirm ist möglich.
- Einfügen von Prozedurenzeilen mit <ESC> + <i>, löschen von Programmzeilen mit <ESC>+<d>

4. ende <Prozedurennummer> eingeben (zum Beispiel ende 24)
- Verschiedene Prozeduren müssen verschiedene Nummern haben, im Zweifelsfall durch zt im Direktmodus Nummern feststellen. Bei Nummerngleichheit wird das alte Programm überschrieben.
- Mit <RETURN> Menü aufrufen.

5. Menü
<CONTROL>+
<r> – Redigieren – unbedingt nötig, falls Prozedur nicht mit <CONTROL> + <b> abgespeichert wird.
<b> – Sichern – Prozedur wird als sequentielles File auf Diskette gesichert. Der Prozedurname wird übernommen, ein File des gleichen Namens überschrieben.
<c> – Einlesen – Prozedur wird in BASIC übersetzt und in den Prozedurenspeicher übernommen. Sie kann danach im Direktmodus zusammen mit weiteren Prozeduren als Prozedurenpaket abgespeichert werden.
<d> – Drucken – Prozedur wird auf den Drucker ausgegeben.
<p> – Plotten – Prozedur wird auf den Plotter VC 1520 ausgegeben.
<g> – Abbruch – Ausstieg aus dem Editor und Rückkehr in den Direktmodus.

Nach <CONTROL> + <c> und der Lernmeldung des Rechners kann die neue Prozedur mit ihrem Namen (und mit einer Längenangabe, falls die Länge variabel gewählt wurde) aufgerufen werden. Mit der STOP-Taste kann jederzeit unterbrochen werden.

6. Drei Möglichkeiten, den Editor zu verlassen:
a) vor Eingabe des Prozedurnamens: zweimal <RETURN> betätigen;
b) während des Schreibens der Prozedur: ls eingeben und zweimal <RETURN> drücken;
c) während des Schreibens der Prozedur: „ende“ eingeben und <CONTROL> + <c> betätigen.

TIPS ZU IGEL

1. Sollten Sie nach dem Befehl „lade“ einmal ein Prozedurenpaket einlesen wollen, das sich nicht auf der eingelegten Diskette befindet, so werden Sie aus dem Programm hinauskatapultiert. Legen Sie dann die Diskette ein, auf der das File „delete“ gespeichert ist und betätigen Sie die Funktionstaste F5.
2. Hindern Sie den Igel daran, eine Prozedur zu Ende zu bringen (mit der <STOP>-Taste jederzeit möglich), so ist manchmal ein zweimaliges Befehlen von „bild“ nötig, um den Igel in die Ausgangsposition zu bringen.
3. Es ist nicht möglich, alle Fähigkeiten von Igel zu demonstrieren. Eigenes Experimentieren ist erwünscht und wird den Anwender bald perfekt machen. Außerdem muß man sich intensiv mit jedem Befehl beschäftigen.
4. Beim Sichern von Prozedurenpaketen empfiehlt sich die Kennung „.pkt“, damit keine Verwechslungen stattfinden.
5. Wählen Sie das Programmpaket „demo.pkt“ □

```
10 rem igel====================p4 <bp>
20 rem (p) commodore welt team    <ho>
30 rem ========================== <ng>
40 rem (c) by k.-d. mueller       <ig>
50 rem                            <pd>
60 rem basic v3.5                 <od>
70 rem plus4 (c16/116 + 64 kb)    <cd>
80 rem disk + drucker (epson-komp. <mn>
90 rem ========================== <jg>
100 graphic1,1:poke55,0:poke56,160
:clr:graphic0:gosub2290           <he>
110 gosub61830:sys4795            <kf>
120 poke4796,112:poke4800,23:poke
4807,252:poke4811,83:sys4293:sys46
39:run1520                        <bn>
121 run1520                       <lc>
130 ed=1:iffa=0thengosub1390:retur
n                                 <gn>
140 dowhileb>32040:b=b-32040:loop:
return                            <eo>
150 x=rdot(0):y=rdot(1):sys50559:b
ox1,0,0,319,199:locate x,y: return <ol>
160 open 1,8,15:input#1,f,f$:iff<>
0thenprint:printf$:forf=1to1000:ne
xt:close1:run2050                 <je>
170 close1:return                 <hc>
180 printchr$(147)                <on>
190 gosub2290:i=3:printrn$he$c4$"P
rozedurname"rf$;:inputc$          <mh>
200 poke 205,1:print:poke202,20:pr
intrn$fl$"<esc> zum Laden!!! "rf$f
o$;                               <kc>
210 getkeyv$:ifv$=chr$(27)thensys4
477,""+c$,3072,0,8:gosub160:printh
e$c4$c4$c4$:goto230               <hh>
220 poke205,1:print:poke202,20:pri
ntb$b5$b4$he$c4$c4$c4$"pr "+c$:ifc
$=""thenrun1520                   <ab>
230 dimle$(20):dimw1$(20):dimw$(20
):n=2:sn$="draw m to"             <el>
240 tu$=":goS130":ko$=":goS140"   <od>
250 va$="=c:":vc$="c=":gosub320   <ce>
260 va$="=z:":vc$=":z=":gosub320  <mj>
270 va$="=iw:":vc$=":iw=":gosub320 <fh>
280 va$="=zl:":vc$=":zl=":gosub320 <gd>
290 va$="=r:":vc$=":r=":gosub320  <op>
300 va$="=zw":vc$=":zw=":gosub320 <ae>
310 forla=1to6:bh$=bh$+bh$(la):br$
=br$+br$(la):next:goto360         <hf>
320 la=la+1:s3=int(rnd(0)*25)+65:s
4=int(rnd(0)*9)                   <bi>
330 bw$(la)=chr$(s3)+str$(s4)     <nc>
340 bh$(la)=bw$(la)+va$:br$(la)=vc
$+bw$(la)                         <ag>
350 return                        <em>
360 w$(i)=str$(i)+bh$:i=i+1:n=n+1 <gb>
370 trap2070:do:gosub1440:gosub599
0                                 <dh>
380 ifa$="warte"thenw1$(i)="for v=
1 to"+str$(a)+tu$+":next":goto840 <ee>
390 ifa$="stifthoch"or a$="sh"then
w1$(i)="rem**pu**":sn$="locate":e$
=a$:goto840                       <nk>
400 ifa$="stiftab"ora$="sa"thenw1$
(i)="rem**pd**":sn$="draw m to":e$
=a$:goto840                       <hm>
410 ifa$="zi"ora$="zeigigel"thenw1
$(i)="fa=0:ed=0":goto840          <km>
420 ifa$="zufallszahl-l"ora$="zz-l
"thenw1$(i)="c=int(rnd(1)*"+str$(a
)+")+1":goto840                   <jk>
430 ifa$="zufallszahl-w"ora$="zz-w
"then w1$(i)="iw=int(rnd(1)*"+str$
(a)+")+1":goto 840                <no>
440 ifa$="aufkurs"ora$="ak"thenw1$
(i)="b="+str$(a):goto840:elseifa$=
"G"thenrun1520                    <nm>
450 ifa$="vi"ora$="versteckigel"th
enw1$(i)="fa=1":goto840           <nk>
460 ifa$="igeltext"thengosub 1360:
goto840                           <ah>
470 ifa$="fuelle"thenw1$(i)="locat
e4;5:paint,rdot(0),rdot(1)":goto84
0                                 <ho>
480 ifa$="invert"ora$="i"thenw1$(i
)="m=-(m=0):sys 4388":goto 840    <cm>
490 ifa$="ls"thenrun180           <jj>
500 ifa$="vorwaerts"ora$="vw"thenw
1$(i)=sn$+str$(a)+f$+";b"+tu$:goto
840                               <dc>
510 ifa$="rueckwaerts"ora$="rw"the
nw1$(i)=sn$+"-("+str$(a)+f$+");b"+
tu$:goto840                       <oj>
520 ifa$="stop.wenn.winkel="ora$="
sww="thenw1$(i)=":do until iw="+st
r$(a):goto840                     <mj>
530 ifa$="stop.wenn.winkel>"ora$="
sww>"thenw1$(i)=":do until iw>"+st
r$(a):goto840                     <aa>
540 ifa$="stop.wenn.winkel<"ora$="
sww<"thenw1$(i)=":do until iw<"+st
r$(a):goto840                     <bn>
550 ifa$="winkel="thenw1$(i)="iw="
+str$(a):goto 840                 <oi>
560 ifa$="var.winkel+"thenw1$(i)="
:":iw$="+iw":goto840              <fa>
570 ifa$="var.winkel*"thenw1$(i)="
:":iw$="*iw":goto840              <mb>
580 ifa$="rechts"ora$="re"thenw1$(
i)="w="+str$(a)+iw$+":b=b+w"+ko$+t
u$:goto840                        <le>
590 ifa$="links"ora$="li"thenw1$(i
)="w=360-("+str$(a)+iw$+"):b=b+w"+
ko$+tu$:goto840                   <ik>
600 ifa$="winkel+"thenw1$(i)="iw=i
w+"+str$(a)+"+zw":goto840         <af>
```

```
610 ifa$="winkel*"thenw1$(i)="iw=i
w*("+str$(a)+"+zw)":goto840         <ok>
620 ifa$="wiederhole"ora$="wh"then
w1$(i)="s1=r:s2=z:z=0:r="+str$(a)+
"+vr:do":goto840                    <om>
630 ifa$="stop.wenn.laenge="ora$="
swl="thenw1$(i)=":do until c="+str
$(a):goto840                        <gg>
640 ifa$="stop.wenn.laenge>"ora$="
swl>"thenw1$(i)=":do until c>"+str
$(a):goto840                        <aj>
650 ifa$="stop.wenn.laenge<"ora$="
swl<"thenw1$(i)=":do until c<"+str
$(a):goto840                        <cb>
660 if a$="ende.wiederhole"ora$="e
ndwh"then1070                       <bb>
670 ifa$=c$thenw1$(i)="loop":goto8
40                                  <ea>
680 ifa$="laenge+"thenw1$(i)="c=c+
"+str$(a)+"+zl":goto840             <no>
690 ifa$="laenge*"thenw1$(i)="c=c*
("+str$(a)+"+zl)":goto840           <ph>
700 ifa$="var.laenge+"thenw1$(i)="
:":f$="+c":goto840                  <la>
710 ifa$="var.laenge*"thenw1$(i)="
:":f$="*c":goto840                  <af>
720 if a$="mitte"thenw1$(i)=sn$+"1
60,100:b=0:w=0":goto840             <ki>
730 ifa$="loeschebild"ora$="lb"the
nw1$(i)="goS150":goto840            <db>
740 ifa$="pause"thenw1$(i)="getkey
 x$":goto840                        <jh>
750 ifa$="laenge="thenw1$(i)="c="+
str$(a):goto840                     <pj>
760 ifa$="aufx"thenw1$(i)=sn$+"160
+"+str$(a)+",rdot(1)"+tu$:goto840   <ie>
770 ifa$="aufy"then w1$(i)=sn$+" r
dot(0),100-"+str$(a)+tu$:goto840    <il>
780 ifa$="radiere"thenw1$(i)="m=0"
:goto840                            <lf>
790 ifa$="farbe"thenw1$(i)="m=1:co
lor1,"+str$(a)+",lu":goto840        <ng>
800 ifa$="rahmen"thenw1$(i)="colo"
+z3$+"4,"+str$(a)+",lu":goto840     <bi>
810 ifa$="luminanz"ora$="lu"thenw1
$(i)="lu="+str$(a):goto840          <mc>
820 ifa$="ende"theni=i+1:n=n+1:got
o 1230                              <am>
830 gosub1200:goto850               <nn>
840 w$(i)=str$(i)+w1$(i):i=i+1:n=n
+1                                  <ba>
850 loop                            <jg>
860 printcl$:sys4940                <nm>
870 ifa<=0ora>50then2050            <pm>
880 k=a:a=10000+(k-1)*20:d=a-4000:
e=a-3000                            <lc>
890 iffl=0thengoto930               <bd>
900 printc4$c4$"delete"d:printc4$c
```

```
4$"delete"e:printc4$c4$"delete"d+1
4000                                <de>
910 printc4$c4$"delete"a"-"a+19:pr
intc4$c4$"rU1520":printhe$          <co>
920 poke 239,5:fori=0to4:poke1319+
i,13:next:end                       <ce>
930 fori=3ton-1:printstr$(a+i-3)+m
id$(w$(i),-(i>9)+3,len(w$(i))):nex
t                                   <mf>
940 printstr$(a-3+n)+br$+":reT:rem
*"+c$+"*"                           <ng>
950 printstr$(e)+"ifa$="+chr$(34)+
c$+chr$(34)+"thEc=a:goS"+str$(a)+"
:w$(i)=str$(i)+";                   <ln>
960 printchr$(34)+"gosub"+str$(a)+
chr$(34)+":i=i+1:goto2120"          <ng>
970 printstr$(d)"ifa$="chr$(34)c$c
hr$(34)"thEw1$(i)="chr$(34)":goS"s
tr$(a)chr$(34);                     <ke>
980 print":g0840"                   <no>
990 print"1050 df$="chr$(34)c$chr$
(34)                                <ea>
1000 printstr$(d+14000)":"k"."b2$c
$:print"goto1040"                   <mn>
1010 if peek(205)<23then1030        <nf>
1020 sys4949:printcl$left$(qr$,5)l
eft$(qd$,7)rn$"zu viele Zeilen!!!"
:getkeyze$:goto2050                 <hb>
1030 poke1319,19:poke1320,17:fori=
1ton+3:poke1320+i,13:next:poke239,
n+5:end                             <nh>
1040 gosub2290:printcl$:sys4949     <gn>
1050 df$="chaos"                    <pm>
1060 printleft$(qr$,4)left$(qd$,4)
rn$df$rf$b3$"gelernt":fori=1to600:
next:run 1520                       <jf>
1070 w1$(i)=+"z=z+1:loOuNz>=r:r=s1
:z=s2":goto840                      <id>
1080 printrn$"x"rf$;:inputx3:print
rn$"y"rf$;:inputy3                  <db>
1090 printrn$"sa (0) oder sh (1)"r
f$;:inputpu                         <ah>
1100 ifx3=1111thenx3=rdot(0):elsex
3=160+x3                            <bb>
1110 ify3=1111theny3=rdot(1):elsey
3=100-y3                            <en>
1120 ifputhenlocatex3,y3:elsedrawm
tox3,y3                             <cb>
1130 goto 2120                      <dn>
1140 :                              <ck>
1150 p%=bl                          <bf>
1160 ifa=8thenscratch""+c$:sys4462
,""+c$,3072,bl*40+3072,a            <ji>
1170 ifa=8thenfory=0tobl:poke1319+
y,13:next:poke239,bl:run 190        <ee>
1180 if a=4thenpoke 4999,7:p%=p%-3
:sys4958,12,4,15,p%:a=s1:goto 1280 <jc>
1190 if a=6thenopen 6,6,6:print#6,
```

```
1:p%=p%-3:poke 4999,0:sys4958,12,6
,15,p%:a=s1:goto1240                <he>
1200 iffa=0thengshapes$(b2),x1-5,y
1-5,4                               <jk>
1210 printc2$rn$"unbekannte Prozed
ur"rf$;:forf=1to1000:next:printchr
$(27)+"p";                          <im>
1220 printchr$(27)+"j";:return      <pk>
1230 s1=a:bl=peek(205):printchr$(2
7)+"c";                             <fb>
1240 forzz=1to37:printzv$;:next:pr
intzv$                              <hn>
1250 print "Ihre Wahl: <ctrl>+....
.............":print                <pc>
1260 printrn$"<c>:'Lernen'<g>:'Sto
ppen'<b>:'Sichern'"rf$:print        <ej>
1270 printrn$"<r>:'Redig.'<d>:'Dru
cker'<p>:'Plotter'"rf$              <ac>
1280 getkeyct$:p1=asc(ct$)          <ig>
1290 ifp1=3thengoto860              <ed>
1300 ifp1=7thenrun1520              <np>
1310 ifp1=2thena=8:goto1140         <al>
1320 ifp1=4thena=4:goto1140         <mp>
1330 ifp1=16thena=6:goto1140        <jf>
1340 ifp1=18thenfory=0tobl:poke131
9+y,13:next:poke239,bl:run190       <db>
1350 goto 1280                      <oa>
1360 inputz$:w1$(i)="char,rdot(0)/
8+1,rdot(1)/8,"+chr$(34)+z$+chr$(3
4)+","+str$(a)                      <pf>
1370 return                         <em>
1380 graphichk,0:sys5376:return     <of>
1390 x1=rdot(0):y1=rdot(1)          <fo>
1400 ifb>=360thenb=b-360:goto1400   <of>
1410 b2=int(b/10):gshapes$(b2),x1-
5,y1-5,4                            <op>
1420 ifed=1thened=0:gshapes$(b2),x
1-5,y1-5,4:locatex1,y1              <dl>
1430 return                         <mf>
1440 poke 19,1:inputa$:poke19,0     <ee>
1450 le=1:me$=a$                    <ao>
1460 if le>20 thena$=me$:a=0:goto1
490:elsele$(le)=mid$(a$,le,1)       <ch>
1470 ifle$(le)<>" "then le=le+1:go
to1460                              <ib>
1480 a1$=right$(a$,len(a$)-le):a=v
al(a1$):a$=left$(a$,(le-1))         <gf>
1490 print:if dz=0thenreturn        <fn>
1500 ifa=0thenprint#3,a$:return     <ij>
1510 ifa<>0thenprint#3,a$;a:return  <pj>
1520 graphic2,1                     <ka>
1530 box0,0,0,319,199:dims$(36):sy
s4940:fornr=0to360step10:circle1,5
,5,4,5,,,nr,120                     <jl>
1540 gosub2290:sshapes$(nr/10),0,0
,10,10:sys4922:next:n=3:s=3:b=0:m=
1                                   <ib>
1550 dimw$(30):dimle$(20):trap2170  <ef>
1560 box 1,0,0,319,199:x1=160:y1=1
00:locatex1,y1:gshapes$(b2),x1-5,y
1-5,4:sys 4949                      <op>
1570 do:gosub1440:iffa=0thengshape
s$(b2),x1-5,y1-5,4:locate x1,y1     <fd>
1580 printbk$;:gosub6990            <oh>
1590 ifa$="ade"thensys4795:sys4660
:clr:pn$="delete":goto2220          <ki>
1600 if a$="basic"thengraphic0:got
o2080:elseifa$="ende"thengraphic0:
end                                 <dh>
1610 if a$="zeige.titel"ora$="zt"t
henpoke 22,35:list 19990-:poke22,2
5:goto2120                          <ao>
1620 ifa$="versteckigel"ora$="vi"t
hen:fa=1:goto2120                   <mm>
1630 ifa$="zeigigel"ora$="zi"thenf
a=0:ed=0:goto2120                   <ah>
1640 if a$<>"igeltext"then1660      <ef>
1650 printrn$+"Text"+rf$;:inputz$:
char,rdot(0)/8+1,rdot(1)/8,z$,a:go
to2120                              <eb>
1660 ifa$="wiederhole"or a$="wh"th
envr=a:goto2120                     <fn>
1670 if a$="fuelle"thenlocate 4;5:
paint,rdot(0),rdot(1):goto 2120     <ob>
1680 ifnot(a$="inhalt"ora$="ih")th
en1700                              <ak>
1690 graphic0:printcl$:directory:g
etkeyw$:printcl$:graphic2,0:poke20
5,19:print:goto2120                 <mg>
1700 ifa$="vorwaertssh"ora$="vwsh"
then locate a;b:goto2120            <fc>
1710 ifa$="rueckwaertssh"ora$="rws
h"thenlocate-a;b:goto2120           <nm>
1720 if a$="ls"thenprintcl$:poke20
5,19:print:goto 2120                <li>
1730 ifa$="rueckwaerts"ora$="rw"th
endrawmto-a;b:goto2120              <hd>
1740 ifa$="vorwaerts"ora$="vw"then
drawmtoa;b:goto2120                 <ej>
1750 ifa$="rechts"ora$="re"thenw=a
:b=b+w:goto2120                     <jf>
1760 ifa$="links"ora$="li"thenw=36
0-a:b=b+w:goto2120                  <og>
1770 if a$="invert"ora$="i"then m=
-(m=0):sys 4388:goto 2120           <mg>
1780 ifa$="xko"thengosub2160:print
b5$b4$c2$x1-160:goto 2120           <ki>
1790 ifa$="yko"thengosub2160:print
b5$b4$c2$100-y1:goto 2120           <nj>
1800 ifa$="schiebbild"thenifa>1the
nifa<4thenq=8192*(a+3):sys(5616)q,
32:goto 2120                        <ie>
1810 ifa$="holbild"thenifa>1thenif
a<4thenq=8192*(a+3):sys(5651)q,32:
goto 2120                           <ao>
1820 ifa$="zeigbild"thenifa>0theni
```

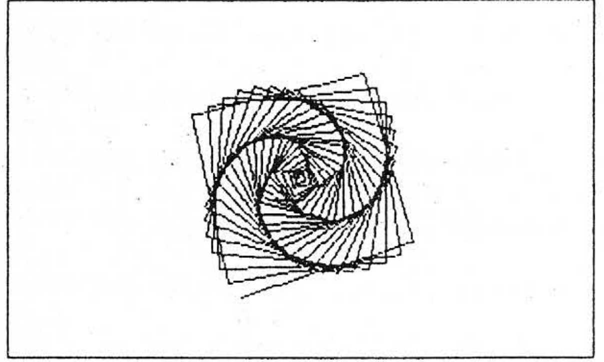
```
fa<4thenq=8*(a+3)+(a=1)*24:poke652
98,q:goto2120                      <bf>
1830 if a$="winkel=" then iw=a:got
o 2120                             <da>
1840 ifa$="winkel+"thenzw=a:goto21
20:elseifa$="laenge+"thenzl=a:goto
2120                               <pk>
1850 ifa$="radiere"thenm=0:goto212
0                                  <cd>
1860 ifa$="luminanz"ora$="lu"thenl
u=a:goto2120                       <el>
1870 ifa$="farbe"then m=1:color1,a
,lu:printbk$;:goto2120             <dp>
1880 ifa$="rahmen"thencolor4,a,lu:
goto2120                           <oi>
1890 ifa$="aufxy"then1080          <gn>
1900 ifa$="kurs"thengosub2150:goto
2120:elseifa$="aufkurs"ora$="ak"th
enb=a:goto2120                     <gk>
1910 ifa$="vergiss"ora$="vg"thenif
a<>0thenfl=1:goto860               <do>
1920 ifa$="loeschefile"theninputlf
$:sys 4940:scratch""+lf$:sys4949:g
oto2120                            <db>
1930 if a$="druckebild"ora$="db"th
enhk=2:sys4940:gosub1380:sys4949:g
oto2120                            <if>
1940 ifa$="druckerein"ora$="dre"th
enopen3,4,7:dz=1:goto2120          <oe>
1950 if a$="druckeraus"ora$="dra"t
hendz=0:close3:goto2120            <nl>
1960 ifa$="ladebild"ora$="bewahreb
ild"then goto 2240                 <ei>
1970 ifa$="lade"ora$="bewahre"then
2190:goto2120                      <hi>
1980 ifnot(a$="loeschebild"ora$="l
b")then2000                        <di>
1990 gosub2160:sys50559:box1,0,0,3
19,199:locatex1,y1:goto2120        <em>
2000 ifa$="mitte"thenlocate160,100
:b=0:w=0:goto2120                  <ne>
2010 ifa$="bild"thensys50559:b2=0:
b=0:box1,0,0,319,199:locate 160,10
0:goto2120                         <cb>
2020 ifa$="edit"thengraphic0:run 1
80                                 <mk>
2030 gosub1200                     <ga>
2040 loop                          <ol>
2050 color0,4:color4,4:color1,1:tr
ap2050:sys4949                     <io>
2060 run1520                       <bn>
2070 hk=1:gosub1380:goto2050       <eg>
2080 printcl$"prozedurnummer";:inp
utu:u1=9980+u*20:u2=10000+u*20-1:p
rintcl$"list"u1"-"u2               <ln>
2090 poke 1319,19:poke 1320,13:pok
e 239,2:end                        <ea>
2100 run1520                       <bc>
2110 end                           <am>
2120 iffa=0thengosub1390:elsegosub
2160                               <gh>
2130 n=n+1                         <hd>
2140 loop                          <le>
2150 rb=b:dountilrb<360:rb=rb-360:
loop:printb$c2$rb:return           <ho>
2160 x1=rdot(0):y1=rdot(1):return  <in>
2170 iffa=0thenifed=0thengshapes$(
b2),x1-5,y1-5,4                    <fm>
2180 resume1570                    <pd>
2190 ifa$="lade"thensys4795:sys466
0:clr:a$="lade"                    <gm>
2200 inputpn$:sys4940:ifa$="bewahr
e"thenscratch""+pn$                <al>
2210 ifa$="bewahre"thensys4672,""+
pn$,5990,0,21500,8:run1520         <jm>
2220 sys4742,""+pn$,5990,8:sys4660
:poke1319,asc("z"):poke1320,asc("t
")                                 <gj>
2230 poke 1321,13:poke239,3:run152
0                                  <hd>
2240 inputpi$:sys 4940             <pj>
2250 ifa$="ladebild"thensys4477,""
+pi$,6144-((a=1)*2048),0,8:sys4949
:gosub160:run1530                  <ln>
2260 sys 4462,""+pi$,6144-((a=1)*2
048),16384,8:sys4949:run1530       <do>
2270 rem nachspann ==============  <nl>
2280 rem * farbcodes/steuercodes * <lo>
2290 c4$=chr$(017):rn$=chr$(018)   <pe>
2300 he$=chr$(019):c3$=chr$(029)   <ol>
2310 fl$=chr$(130):fo$=chr$(132)   <mp>
2320 bk$=chr$(144):c2$=chr$(145)   <cc>
2330 rf$=chr$(146):cl$=chr$(147)   <ek>
2340 rem *** zeichensatz/graphik * <fg>
2350 z3$=chr$(164):zv$=chr$(192)   <ck>
2360 rem ********* zeichenfolgen * <hi>
2370 qd$="":qr$="":forzz=1 to 40   <al>
2380 qd$=qd$+c4$:qr$=qr$+c3$       <bj>
2390 nextzz                        <ap>
2400 b$=chr$(32):b2$=b$+b$         <fk>
2410 b3$=b2$+b$:b4$=b3$+b$         <pi>
2420 b5$=b4$+b$:b$=b5$+b5$         <ob>
2430 return                        <jn>
5990 rem**neue befehle(programmmod
us)***                             <ba>
6989 return                        <fo>
6990 rem***neue befehle (direktmod
us)***                             <oc>
7989 return                        <dg>
19990 rem*****befehlssatz*****     <fl>
60000 data00,00,00,58,00,00,00,00,
 88                                <bp>
60010 data00,00,53,00,00,00,ad,09,
 265                               <bm>
60020 dataff,29,02,f0,03,20,60,10,
 685                               <oa>
```

```
60030 data2c,d8,07,10,0e,ad,01,fd,
 724                                    <ce>
60040 data8d,d4,07,10,06,20,95,ea,
 797                                    <ok>
60050 data20,5b,ea,20,e4,e3,ad,09,
 1026                                   <ok>
60060 dataff,29,02,f0,28,8d,09,ff,
 983                                    <bl>
60070 data2c,0b,ff,a9,cc,50,1b,6c,
 898                                    <hm>
60080 data12,03,20,bf,cf,20,cd,ce,
 894                                    <nm>
60090 dataa5,fb,48,a9,00,85,fb,08,
 1049                                   <bm>
60100 data58,20,11,db,28,68,85,fb,
 884                                    <he>
60110 dataa9,a1,8d,0b,ff,4c,be,fc,
 1255                                   <jd>
60120 dataad,1c,ff,29,01,d0,39,ad,
 936                                    <kd>
60130 data1d,ff,c9,a4,b0,2e,24,83,
 1038                                   <pc>
60140 data50,52,a9,08,8d,14,ff,ad,
 928                                    <oh>
60150 data06,ff,29,df,a8,ad,07,ff,
 1128                                   <pa>
60160 data29,ef,aa,ad,12,ff,0d,fa,
 1159                                   <di>
60170 data07,48,ad,1d,ff,c9,a4,90,
 1045                                   <mk>
60180 dataf9,68,8d,12,ff,8c,06,ff,
 1168                                   <el>
60190 data8e,07,ff,60,c9,cc,90,24,
 1085                                   <la>
60200 dataa6,83,f0,20,10,08,ad,07,
 773                                    <kl>
60210 dataff,09,10,8d,07,ff,ad,06,
 862                                    <jg>
60220 dataff,09,20,8d,06,ff,ad,12,
 889                                    <kj>
60230 dataff,29,fb,8d,12,ff,ad,fb,
 1385                                   <ml>
60240 data07,8d,14,ff,60,78,a9,10,
 824                                    <ek>
60250 data8d,13,03,8d,15,03,a9,d5,
 710                                    <oh>
60260 data8d,14,03,58,60,a5,c6,c9,
 912                                    <lc>
60270 data04,d0,0d,a9,06,8d,59,10,
 646                                    <hf>
60280 dataa9,07,8d,6b,10,8d,8e,10,
 739                                    <ml>
60290 datac9,05,d0,0d,a9,81,8d,59,
 955                                    <je>
60300 data10,a9,84,8d,6b,10,8d,8e,
 864                                    <fh>
60310 data10,c9,06,d0,0d,a9,a1,8d,
 915                                    <cb>
60320 data59,10,a9,a4,8d,6b,10,8d,
 843                                    <jb>
60330 data8e,10,c9,03,d0,11,a9,79,
 877                                    <kn>
60340 data8d,59,10,a9,7c,8d,6b,10,
 803                                    <bc>
60350 data8d,8e,10,a9,40,85,c6,4c,
 939                                    <al>
60360 data0e,10,11,78,a9,00,85,e0,
 693                                    <pm>
60370 dataa9,20,85,e1,a0,00,b1,e0,
 1120                                   <mp>
60380 data49,ff,91,e0,c8,d0,f7,e6,
 1582                                   <ej>
60390 datae1,a5,e1,c9,40,d0,ef,60,
 1423                                   <aj>
60400 data49,29,20,91,94,20,2c,93,
 662                                    <nn>
60410 dataa5,62,85,af,a5,63,85,b0,
 1144                                   <if>
60420 dataa5,61,85,ab,20,91,94,20,
 923                                    <al>
60430 data14,93,20,e4,9d,84,da,85,
 1067                                   <jf>
60440 datadb,20,91,94,20,d2,9d,a9,
 1112                                   <mb>
60450 data02,85,ac,86,ae,60,20,42,
 809                                    <mp>
60460 data11,85,ad,a9,da,a6,14,a4,
 1060                                   <ei>
60470 data15,20,d8,ff,60,20,42,11,
 735                                    <jj>
60480 dataa9,00,85,ad,a6,da,a4,db,
 1242                                   <ae>
60490 data20,d5,ff,60,20,42,11,a9,
 880                                    <mc>
60500 data01,85,ad,a9,00,20,d5,ff,
 976                                    <ep>
60510 data60,ea,20,91,94,20,84,9d,
 976                                    <bb>
60520 dataa9,00,a8,85,d2,85,22,86,
 981                                    <oj>
60530 datad3,20,91,94,20,84,9d,86,
 991                                    <ge>
60540 data23,20,91,94,20,84,9d,ea,
 915                                    <mg>
60550 dataea,20,b0,04,91,d2,c8,d0,
 1209                                   <ji>
60560 dataf6,e6,23,e6,d3,ca,d0,ef,
 1601                                   <lg>
60570 data60,20,91,94,20,84,9d,e0,
 966                                    <cm>
60580 data00,d0,0a,20,91,94,20,84,
 707                                    <hp>
60590 data9d,8e,14,ff,60,e0,01,d0,
 1103                                   <mi>
60600 data0a,20,91,94,20,84,9d,8e,
 798                                    <if>
```

```
60610 data12,ff,60,4c,a1,94,a5,2d,
 964                                <ca>
60620 data85,dc,a5,2e,85,dd,60,a5,
 1179                               <ld>
60630 datadc,85,2d,a5,dd,85,2e,60,
 1059                               <fm>
60640 dataa5,2b,85,e8,a5,2c,85,e9,
 1148                               <oo>
60650 data38,a5,2d,e9,02,85,2b,a5,
 842                                <ee>
60660 data2e,e9,00,85,2c,60,a5,e8,
 949                                <pe>
60670 data85,2b,a5,e9,85,2c,60,a9,
 1016                               <kf>
60680 data2a,8d,06,03,a9,12,8d,07,
 527                                <kh>
60690 data03,60,48,ad,43,05,d0,fb,
 875                                <dk>
60700 data68,4c,6e,8b,a9,01,8d,02,
 742                                <ef>
60710 data40,20,18,88,20,4c,88,60,
 596                                <pa>
60720 data20,91,94,20,2c,93,a5,62,
 811                                <id>
60730 data85,af,a5,63,85,b0,a5,61,
 1143                               <je>
60740 data85,ab,20,91,94,20,d2,9d,
 1028                               <an>
60750 data20,3d,8a,a5,5f,85,da,a5,
 1007                               <dn>
60760 data60,85,db,20,91,94,20,d2,
 1015                               <gb>
60770 data9d,86,ae,20,3d,8a,18,a5,
 885                                <gg>
60780 data5f,69,0a,85,dc,a5,60,69,
 929                                <je>
60790 data00,85,dd,a9,02,85,ac,85,
 963                                <bn>
60800 dataad,a9,da,4c,32,13,20,91,
 882                                <me>
60810 data94,20,2c,93,a5,62,85,af,
 942                                <ao>
60820 dataa5,63,85,b0,a5,61,85,ab,
 1139                               <kf>
60830 data20,91,94,20,d2,9d,86,ae,
 1032                               <nl>
60840 data20,3d,8a,a5,5f,85,da,a5,
 1007                               <ha>
60850 data60,85,db,a9,02,85,ac,a9,
 1093                               <nc>
60860 data00,85,ad,a6,da,a4,db,20,
 1105                               <gd>
60870 datad5,ff,60,a9,70,85,14,a9,
 1167                               <oj>
60880 data17,85,15,20,3d,8a,a9,fc,
 829                                <jb>
60890 data85,14,a9,53,85,15,20,d2,
 801                                <en>
60900 data12,60,a5,5f,a6,60,85,24,
 805                                <ee>
60910 data86,25,20,3d,8a,90,15,a0,
 727                                <on>
60920 data01,20,d1,04,88,aa,d0,05,
 765                                <nc>
60930 data20,d1,04,f0,07,20,d1,04,
 737                                <nb>
60940 data85,5f,86,60,a5,24,38,e5,
 944                                <ko>
60950 data5f,aa,a5,25,e5,60,a8,b0,
 1136                               <gl>
60960 data1f,8a,18,65,2d,85,2d,98,
 669                                <hp>
60970 data65,2e,85,2e,a0,00,20,d1,
 727                                <el>
60980 data04,91,24,c8,d0,f8,e6,60,
 1167                               <hj>
60990 datae6,25,a5,2e,c5,25,b0,ee,
 1126                               <eh>
61000 data20,18,88,a5,22,a6,23,18,
 616                                <ki>
61010 data69,02,85,2d,90,01,e8,86,
 796                                <le>
61020 data2e,60,a6,dc,a4,dd,20,d8,
 1161                               <np>
61030 dataff,60,a0,00,98,99,00,20,
 848                                <fn>
61040 data99,40,21,99,80,22,c8,c0,
 957                                <po>
61050 data18,d0,f2,60,ad,06,ff,29,
 1045                               <bh>
61060 dataef,8d,06,ff,60,ad,06,ff,
 1171                               <bg>
61070 data09,10,8d,06,ff,60,20,91,
 700                                <hj>
61080 data94,20,84,9d,86,23,20,91,
 815                                <em>
61090 data94,20,84,9d,86,ae,20,91,
 954                                <ak>
61100 data94,20,84,9d,86,d3,20,91,
 991                                <el>
61110 data94,20,84,9d,86,d4,18,d8,
 1055                               <nh>
61120 dataa9,01,85,ac,85,ab,a9,07,
 955                                <fm>
61130 data85,ad,20,c0,ff,a0,00,a2,
 1107                               <of>
61140 data01,20,c9,ff,a9,0d,20,d2,
 913                                <kc>
61150 dataff,a6,d4,a0,78,98,85,22,
 1232                               <jd>
61160 dataa0,00,a5,d3,20,d2,ff,20,
 1065                               <go>
61170 datab0,04,29,3f,85,d0,20,b0,
 833                                <mp>
61180 data04,29,40,85,d1,18,65,d1,
 785                                <ea>
```

```
61190 data65,d0,85,d0,20,b0,04,49,
 935                              <ba>
61200 data7f,29,20,85,d1,18,65,d1,
 876                              <ke>
61210 data65,d0,85,d2,20,b0,04,29,
 905                              <in>
61220 data80,c9,80,d0,17,a9,12,20,
 907                              <ej>
61230 datad2,ff,ea,ea,ea,ea,ea,a5,
 1800                             <ch>
61240 datad2,20,d2,ff,a9,92,20,d2,
 1264                             <gg>
61250 dataff,18,90,05,a5,d2,20,d2,
 1045                             <ne>
61260 dataff,c8,c0,28,30,b1,a9,0d,
 1094                             <mo>
61270 data20,d2,ff,18,a5,22,69,28,
 865                              <nh>
61280 data85,22,a5,23,69,00,85,23,
 640                              <hm>
61290 dataca,d0,95,a9,0d,20,d2,ff,
 1238                             <dl>
61300 data20,cc,ff,a9,01,20,c3,ff,
 1143                             <bb>
61310 data60,ff,c4,28,49,29,aa,22,
 905                              <ha>
61320 fort= 4096 to 5145 step8:p=0 <md>
61330 fori=0to7:reada$:a=dec(a$):p
=p+a:poket+i,a:nexti              <ij>
61340 readr:ifp=rthen61360        <fd>
61350 print"Pruefsummenfehler in z
eile"peek(63)+peek(64)*256:gosub61
910:end                           <jn>
61360 next                        <mb>
61370 dataea,ea,ea,ea,ea,ea,ea,ea,
 1872                             <em>
61380 dataea,ea,ea,ea,ea,ea,ea,a2,
 1800                             <fh>
61390 data18,8e,60,15,a9,05,48,a2,
 691                              <jf>
61400 data04,ea,ea,a9,00,a0,20,85,
 966                              <pk>
61410 datafa,84,fb,68,48,a0,01,20,
 1002                             <lk>
61420 databa,ff,a9,00,20,bd,ff,20,
 1118                             <aj>
61430 datac0,ff,68,48,aa,20,c9,ff,
 1281                             <ae>
61440 dataa9,1b,20,d2,ff,a9,31,20,
 943                              <pm>
61450 datad2,ff,20,5f,15,a9,1b,20,
 841                              <ih>
61460 datad2,ff,a9,32,20,d2,ff,20,
 1213                             <ea>
61470 datacc,ff,68,4c,c3,ff,00,00,
 1089                             <gc>
61480 data00,00,00,00,00,00,16,a2,
 184                              <ma>
61490 data18,a9,0d,20,d2,ff,a0,05,
 868                              <ml>
61500 datab9,8f,15,20,d2,ff,88,10,
 998                              <ac>
61510 dataf7,a9,27,8d,5e,15,a0,00,
 871                              <ic>
61520 data20,95,15,a5,fa,18,69,08,
 754                              <bn>
61530 data90,02,e6,fb,85,fa,ce,5e,
 1310                             <cm>
61540 data15,10,eb,ca,10,d3,60,01,
 798                              <gl>
61550 data40,04,2a,1b,09,8a,48,98,
 508                              <jf>
61560 data48,a9,80,8d,d2,15,a2,07,
 910                              <cp>
61570 dataa0,00,b1,fa,2d,d2,15,38,
 919                              <cj>
61580 datad0,01,18,3e,56,15,c8,c0,
 794                              <li>
61590 data08,d0,ef,4e,d2,15,ca,10,
 982                              <kg>
61600 datae7,a2,07,bd,56,15,20,d2,
 938                              <ia>
61610 dataff,ea,ea,ea,a9,00,9d,56,
 1369                             <ci>
61620 data15,ca,10,ef,68,a8,68,aa,
 1024                             <je>
61630 data60,00,ea,ea,ea,ea,ea,ea,
 1500                             <ak>
61640 dataea,ea,ea,ea,ea,ea,ea,ea,
 1872                             <jk>
61650 dataea,ea,ea,ea,ea,ea,ea,ea,
 1872                             <la>
61660 dataea,ea,ea,ea,ea,ea,ea,ea,
 1872                             <kh>
61670 data20,d2,9d,a0,00,98,85,d0,
 1052                             <hp>
61680 dataa9,20,85,d1,a5,14,85,d2,
 1071                             <ic>
61690 dataa5,15,85,d3,b1,d0,91,d2,
 1270                             <mj>
61700 datac8,d0,f9,e6,d1,e6,d3,ca,
 1739                             <of>
61710 datad0,f2,60,20,d2,9d,a9,00,
 1114                             <ei>
61720 data85,d2,a9,20,85,d3,a5,14,
 1073                             <jn>
61730 data85,22,a5,15,85,23,ea,a0,
 915                              <mc>
61740 data00,b1,22,20,b0,04,91,d2,
 778                              <hi>
61750 datac8,d0,f6,e6,23,e6,d3,ca,
 1562                             <fm>
61760 datad0,ef,60,a2,01,bd,00,0f,
 910                              <id>
61770 fort= 5376 to 5695 step8:p=0 <bn>
61780 fori=0to7:reada$:a=dec(a$):p
```

```
=p+a:poket+i,a:nexti                    <fe>
61790 readr:ifp=rthen61810              <mm>
61800 print"Pruefsummenfehler in z
eile"peek(63)+peek(64)*256:gosub61
910:end                                 <po>
61810 next                              <ei>
61820 return                            <gi>
61830 gosub 61900                       <mb>
61840 color0,4:color4,4:color1,1:p
rintchr$(14)                            <lg>
61850 printcl$                          <ep>
61860 fori=1to8:keyi,"":next:key4,
"run1520"+chr$(13)                      <lo>
61870 key5,"pn$="+chr$(34)+"delete
"+chr$(34)+":goto2220"+chr$(13)         <fo>
61880 gosub60000:poke4796,96:poke4
800,234:poke4807,48:poke 4811,242       <ik>
61890 return                            <pf>
61900 poke 65286,peek(65286) and 2
39:return                               <dd>
61910 poke 65286,peek(65286) or 16
:return                                 <nl>
61920 rem igel==================p4      <gn>
61930 rem 60671 bytes memory            <ek>
61940 rem 15907 bytes programm          <an>
61950 rem (im urzustand)                <ln>
61960 rem                               <ie>
61970 rem achtung !!!                   <on>
61980 rem programm darf nicht           <mj>
61990 rem umnummeriert werden           <ji>
62000 rem ======================        <mo>
```



DELETE FUER IGELGRAFIK

```
5990 rem *neue befehle (programmmo
dus)*                                   <dm>
6989 return                             <fo>
6990 rem *neue befehle (direktmodu
s)*                                     <pe>
7989 return                             <dg>
19990 *****befehlssatz*****             <kn>
```

((BU:)) Programm DELETE fuer IGEL-Grafik

```
5990 rem demo.pkt==============p4      <nj>
6000 ifa$="quadrat"thenw$(i)=str$(
i)+":goS 10000"  :goto840               <nc>
6020 ifa$="kiste5"thenw$(i)=str$(i
)+":goS 10020"   :goto840               <ol>
6040 ifa$="stern5"thenw$(i)=str$(i
)+":goS 10040"   :goto840               <pd>
6060 ifa$="dreieck"thenw$(i)=str$(
i)+":goS 10060"  :goto840               <pe>
6080 ifa$="drehquadrat"thenw$(i)=s
tr$(i)+":goS 10080"       :goto84
0                                       <gp>
6100 ifa$="mitte"thenw$(i)=str$(i)
+":goS 10100"    :goto840               <oo>
6120 ifa$="demodrehquadrat"thenw$(
i)=str$(i)+":goS 10120"   :goto84
0                                       <jl>
6140 ifa$="stern"thenw$(i)=str$(i)
+":goS 10140"    :goto840               <ko>
6160 ifa$="demostern"thenw$(i)=str
$(i)+":goS 10160"         :goto84
0                                       <ab>
6180 ifa$="stern2"thenw$(i)=str$(i
)+":goS 10180"   :goto840               <ed>
6200 ifa$="demostern2"thenw$(i)=st
r$(i)+":goS 10200"        :goto84
0                                       <kp>
6220 ifa$="rechteck1-5"thenw$(i)=s
tr$(i)+":goS 10220"       :goto84
0                                       <bn>
6240 ifa$="rechteck1-3"thenw$(i)=s
tr$(i)+":goS 10240"       :goto84
0                                       <kp>
6260 ifa$="rechteckrot."thenw$(i)=
str$(i)+":goS 10260"      :goto84
0                                       <bj>
6280 ifa$="drehquadrat2"thenw$(i)=
str$(i)+":goS 10280"      :goto84
0                                       <ho>
6300 ifa$="fassade"thenw$(i)=str$(
i)+":goS 10300"  :goto840               <lf>
6320 ifa$="gehvor"thenw$(i)=str$(i
)+":goS 10320"   :goto840               <bg>
6340 ifa$="fassaden"thenw$(i)=str$
(i)+":goS 10340" :goto840               <lo>
6360 ifa$="rbogen"thenw$(i)=str$(i
)+":goS 10360"   :goto840               <dh>
6380 ifa$="blatt"thenw$(i)=str$(i)
+":goS 10380"    :goto840               <fo>
6400 ifa$="dolde"thenw$(i)=str$(i)
+":goS 10400"    :goto840               <fj>
6420 ifa$="stiel"thenw$(i)=str$(i)
+":goS 10420"    :goto840               <lm>
6440 ifa$="blume"thenw$(i)=str$(i)
+":goS 10440"    :goto840               <kf>
6460 ifa$="garten"thenw$(i)=str$(i
)+":goS 10460"   :goto840               <bo>
6480 ifa$="blumenmotiv"thenw$(i)=s
```

```
tr$(i)+":goS 10480"          :goto84
0                                   <gl>
6500 ifa$="demo1"thenw$(i)=str$(i)
+":goS 10500"     :goto840      <nc>
6520 ifa$="demo2"thenw$(i)=str$(i)
+":goS 10520"     :goto840      <ih>
6540 ifa$="demo3"thenw$(i)=str$(i)
+":goS 10540"     :goto840      <ap>
6560 ifa$="demo4"thenw$(i)=str$(i)
+":goS 10560"     :goto840      <fk>
6580 ifa$="demo"thenw$(i)=str$(i)+
":goS 10580"      :goto840      <jm>
6600 ifa$="wachsquadrat"thenw$(i)=
str$(i)+":goS 10600"         :goto84
0                                   <nf>
6620 ifa$="wachsfassade"thenw$(i)=
str$(i)+":goS 10620"         :goto84
0                                   <ah>
6989 return                         <fo>
6990 rem***neue befehle (direktmod
us)***                              <oc>
7000 ifa$="quadrat"thenc=a:gosub 1
0000:goto2120                       <mg>
7020 ifa$="kiste5"thenc=a:gosub 10
020:goto2120                        <fg>
7040 ifa$="stern5"thenc=a:gosub 10
040:goto2120                        <cj>
7060 ifa$="dreieck"thenc=a:gosub 1
0060:goto2120                       <dc>
7080 ifa$="drehquadrat"thenc=a:gos
ub 10080:goto2120                   <jo>
7100 ifa$="mitte"thenc=a:gosub 101
00:goto2120                         <lg>
7120 ifa$="demodrehquadrat"thenc=a
:gosub 10120:goto2120               <ag>
7140 ifa$="stern"thenc=a:gosub 101
40:goto2120                         <mj>
7160 ifa$="demostern"thenc=a:gosub
 10160:goto2120                     <pj>
7180 ifa$="stern2"thenc=a:gosub 10
180:goto2120                        <ad>
7200 ifa$="demostern2"thenc=a:gosu
b 10200:goto2120                    <nc>
7220 ifa$="rechteck1-5"thenc=a:gos
ub 10220:goto2120                   <dn>
7240 ifa$="rechteck1-3"thenc=a:gos
ub 10240:goto2120                   <fa>
7260 ifa$="rechteckrot."thenc=a:go
sub 10260:goto2120                  <op>
7280 ifa$="drehquadrat2"thenc=a:go
sub 10280:goto2120                  <jd>
7300 ifa$="fassade"thenc=a:gosub 1
0300:goto2120                       <ae>
7320 ifa$="gehvor"thenc=a:gosub 10
320:goto2120                        <bf>
7340 ifa$="fassaden"thenc=a:gosub
10340:goto2120                      <kd>
7360 ifa$="rbogen"thenc=a:gosub 10
360:goto2120                        <hi>
7380 ifa$="blatt"thenc=a:gosub 103
80:goto2120                         <ej>
7400 ifa$="dolde"thenc=a:gosub 104
00:goto2120                         <fc>
7420 ifa$="stiel"thenc=a:gosub 104
20:goto2120                         <ng>
7440 ifa$="blume"thenc=a:gosub 104
40:goto2120                         <ml>
7460 ifa$="garten"thenc=a:gosub 10
460:goto2120                        <kl>
7480 ifa$="blumenmotiv"thenc=a:gos
ub 10480:goto2120                   <hp>
7500 ifa$="demo1"thenc=a:gosub 105
00:goto2120                         <hb>
7520 ifa$="demo2"thenc=a:gosub 105
20:goto2120                         <dg>
7540 ifa$="demo3"thenc=a:gosub 105
40:goto2120                         <de>
7560 ifa$="demo4"thenc=a:gosub 105
60:goto2120                         <gc>
7580 ifa$="demo"thenc=a:gosub 1058
0:goto2120                          <bd>
7600 ifa$="wachsquadrat"thenc=a:go
sub 10600:goto2120                  <if>
7620 ifa$="wachsfassade"thenc=a:go
sub 10620:goto2120                  <bi>
7989 return                         <dg>
10000 :                             <ki>
10001 draw m to 0+c;b               <ig>
10002 w= 90:b=b+w:if b>360thenb=b-
360                                 <fh>
10003 draw m to 0+c;b               <ii>
10004 w= 90:b=b+w:if b>360thenb=b-
360                                 <dd>
10005 draw m to 0+c;b               <hc>
10006 w= 90:b=b+w:if b>360thenb=b-
360                                 <dp>
10007 draw m to 0+c;b               <gm>
10008 w= 90:b=b+w:if b>360thenb=b-
360                                 <ik>
10009 return                        <ao>
10020 c= 10                         <ch>
10021 :gosub 10000                  <cl>
10022 c= 20                         <ei>
10023 :gosub 10000                  <ci>
10024 c= 30                         <gk>
10025 :gosub 10000                  <cn>
10026 c= 40                         <im>
10027 :gosub 10000                  <co>
10028 c= 50                         <la>
10029 :gosub 10000                  <cp>
10030 return                        <di>
10040 :                             <pi>
10041 r1= 5                         <on>
10042 z 1=0:do                      <ee>
10043 draw m to 0+c;b               <nc>
10044 w= 144:b=b+w:if b>360thenb=b
```

```
-360                                   <mg>
10045 z 1=z 1+1:loopuntilz 1>=r 1      <oc>
10046 return                           <fi>
10060 :                                <ca>
10061 r1= 3                            <co>
10062 z 1=0:do                         <eh>
10063 draw m to 0+c;b                  <mg>
10064 w= 120:b=b+w:if b>360thenb=b
-360                                   <bn>
10065 z 1=z 1+1:loopuntilz 1>=r 1      <io>
10066 return                           <ia>
10080 :                                <ej>
10081 do until b> 359                  <mi>
10082 c= 50                            <bd>
10083 :gosub 10000                     <el>
10084 w= 0+iw:b=b+w:if b>360thenb=
b-360                                  <jp>
10085 loop:c=0                         <mc>
10086 return                           <kj>
10100 rem**pu**                         <op>
10101 locate160+((rgr(3)=3orrgr(4)
=4)*80),100:b=0:w=0                    <nh>
10102 rem**pd**                         <km>
10103 for v=1 to 2000:next             <dm>
10104 x=rdot(0):y=rdot(1):scnclr:l
ocate x,y                              <kc>
10105 return                           <mp>
10120 iw= 45                           <kj>
10121 :gosub 10080                     <im>
10122 :gosub 10100                     <ac>
10123 iw= 90                           <ao>
10124 :gosub 10080                     <ap>
10125 :gosub 10100                     <ic>
10126 iw= 15                           <ge>
10127 :gosub 10080                     <ij>
10128 :gosub 10100                     <pn>
10129 iw= 50                           <mh>
10130 :gosub 10080                     <ac>
10131 return                           <ad>
10140 :                                <mb>
10141 do until b> 359                  <ko>
10142 draw m to 50;b                   <ak>
10143 w= 0+iw:b=b+w:if b>360thenb=
b-360                                  <ko>
10144 loop:c=0                         <cd>
10145 return                           <bp>
10160 iw= 135                          <hc>
10161 :gosub 10140                     <he>
10162 :gosub 10100                     <og>
10163 iw= 144                          <dd>
10164 :gosub 10140                     <op>
10165 :gosub 10100                     <gg>
10166 iw= 150                          <no>
10167 :gosub 10140                     <hb>
10168 :gosub 10100                     <pj>
10169 iw= 160                          <kf>
10170 :gosub 10140                     <ac>
10171 :gosub 10100                     <hl>
10172 return                           <ff>
10180 :                                <bb>
10181 :                                <bd>
10182 do until b> 359                  <ej>
10183 draw m to 0+c;b                  <do>
10184 w= 0+iw:b=b+w:if b>360thenb=
b-360                                  <lm>
10185 loop:c=0                         <ep>
10186 return                           <hb>
10200 iw= 144                          <gn>
10201 c= 100                           <bg>
10202 :gosub 10180                     <ji>
10203 :gosub 10100                     <ak>
10204 iw= 135                          <el>
10205 c= 50                            <ll>
10206 :gosub 10180                     <jk>
10207 :gosub 10100                     <ai>
10208 iw= 160                          <eh>
10209 c= 20                            <pf>
10210 :gosub 10180                     <je>
10211 return                           <kc>
10220 :                                <ga>
10221 r1= 2                            <cl>
10222 z 1=0:do                         <ep>
10223 draw m to 1*c;b                  <ie>
10224 w= 90:b=b+w:if b>360thenb=b-
360                                    <bl>
10225 draw m to 5*c;b                  <me>
10226 w= 90:b=b+w:if b>360thenb=b-
360                                    <oh>
10227 z 1=z 1+1:loopuntilz 1>=r 1      <la>
10228 return                           <me>
10240 :                                <ik>
10241 r1= 2                            <hp>
10242 z 1=0:do                         <jo>
10243 draw m to 1*c;b                  <kd>
10244 w= 90:b=b+w:if b>360thenb=b-
360                                    <bc>
10245 draw m to 3*c;b                  <mb>
10246 w= 90:b=b+w:if b>360thenb=b-
360                                    <oo>
10247 z 1=z 1+1:loopuntilz 1>=r 1      <lg>
10248 return                           <oo>
10260 r2= 6                            <li>
10261 z 2=0:do                         <gh>
10262 :gosub 10220                     <dn>
10263 w= 60:b=b+w:if b>360thenb=b-
360                                    <ep>
10264 z 2=z 2+1:loopuntilz 2>=r 2      <ba>
10265 return                           <ba>
10280 :                                <nk>
10281 z 2=0:do                         <ga>
10282 :gosub 10000                     <co>
10283 w= 0+iw:b=b+w:if b>360thenb=
b-360                                  <dl>
10284 z 2=z 2+1:loopuntilz 2>=r 2      <me>
10285 return                           <di>
10300 :                                <ac>
```

```
10301 :gosub 10000                          <je>
10302 draw m to 0+c;b                       <le>
10303 w= 30:b=b+w:if b>360thenb=b-
360                                         <di>
10304 :gosub 10060                          <bn>
10305 w=360-( 30):b=b+w:ifb>360the
nb=b-360                                    <po>
10306 draw m to-( 0+c);b                    <ni>
10307 return                                <ge>
10320 :                                     <ck>
10321 rem**pu**                             <jk>
10322 w= 90:b=b+w:if b>360thenb=b-
360                                         <ko>
10323 locate 10+c;b                         <gl>
10324 w=360-( 90):b=b+w:ifb>360the
nb=b-360                                    <dd>
10325 rem**pd**                             <fk>
10326 return                                <ik>
10340 c=-100                                <en>
10341 :gosub 10320                          <ni>
10342 c= 10                                 <fk>
10343 :gosub 10300                          <nj>
10344 :gosub 10320                          <fm>
10345 c= 20                                 <im>
10346 :gosub 10300                          <fl>
10347 :gosub 10320                          <np>
10348 c= 30                                 <ma>
10349 :gosub 10300                          <ni>
10350 :gosub 10320                          <fn>
10351 return                                <ln>
10360 :                                     <hl>
10361 r1= 9                                 <an>
10362 z 1=0:do                              <le>
10363 draw m to 0+c;b                       <al>
10364 w= 10:b=b+w:if b>360thenb=b-
360                                         <cm>
10365 z 1=z 1+1:loopuntilz 1>=r 1           <ed>
10366 return                                <nl>
10380 :gosub 10360                          <jg>
10381 w= 90:b=b+w:if b>360thenb=b-
360                                         <ij>
10382 :gosub 10360                          <jb>
10383 w= 90:b=b+w:if b>360thenb=b-
360                                         <cn>
10384 return                                <pp>
10400 r2= 3                                 <ho>
10401 z 2=0:do                              <ig>
10402 :gosub 10380                          <ih>
10403 w= 60:b=b+w:if b>360thenb=b-
360                                         <pl>
10404 z 2=z 2+1:loopuntilz 2>=r 2           <pe>
10405 return                                <cj>
10420 :                                     <pd>
10421 draw m to 7*c;b                       <mo>
10422 :gosub 10380                          <ib>
10423 draw m to 14*c;b                      <pm>
10424 :gosub 10400                          <ie>
10425 return                                <fb>
10440 :                                     <bl>
10441 :gosub 10420                          <pc>
10442 :gosub 10400                          <gp>
10443 draw m to-(21*c);b                    <im>
10444 return                                <hh>
10460 c=-60                                 <ij>
10461 :gosub 10320                          <je>
10462 c= 1                                  <nb>
10463 :gosub 10440                          <jj>
10464 c= 30                                 <ph>
10465 :gosub 10320                          <ja>
10466 c= 2                                  <pc>
10467 :gosub 10440                          <kd>
10468 c= 30                                 <dg>
10469 :gosub 10320                          <ji>
10470 c= 3                                  <bd>
10471 :gosub 10440                          <kf>
10472 return                                <ko>
10480 :                                     <gk>
10481 b=0:   do until b> 359                <io>
10482 :gosub 10440                          <bk>
10483 w= 0+iw:b=b+w:if b>360thenb=
b-360                                       <ab>
10484 loop                                  <ce>
10485 return                                <mi>
10500 c= 50                                 <fa>
10501 :gosub 10000                          <dj>
10502 :gosub 10100                          <mb>
10503 :gosub 10020                          <ea>
10504 :gosub 10100                          <mc>
10505 c= 80                                 <ke>
10506 :gosub 10040                          <mh>
10507 :gosub 10100                          <ec>
10508 c= 40                                 <mm>
10509 :gosub 10060                          <ej>
10510 :gosub 10100                          <ln>
10511 return                                <po>
10520 :gosub 10120                          <ko>
10521 :gosub 10100                          <ck>
10522 :gosub 10160                          <lh>
10523 :gosub 10100                          <cl>
10524 c= 20                                 <mj>
10525 :gosub 10220                          <db>
10526 :gosub 10100                          <kf>
10527 c= 30                                 <pl>
10528 :gosub 10240                          <lc>
10529 :gosub 10100                          <co>
10530 c= 20                                 <cj>
10531 :gosub 10260                          <do>
10532 :gosub 10100                          <km>
10533 return                                <ck>
10540 iw= 72                                <fj>
10541 c= 50                                 <oa>
10542 :gosub 10280                          <mb>
10543 :gosub 10100                          <cn>
10544 :gosub 10340                          <lo>
10545 :gosub 10100                          <cg>
10546 c= 10                                 <ci>
```

DEMO ZUR IGELGRAFIK

```
10547 :gosub 10360                     <dk>
10548 :gosub 10100                     <ke>
10549 :gosub 10380                     <dn>
10550 :gosub 10100                     <kj>
10551 return                           <eo>
10560 c= 2                             <ol>
10561 :gosub 10400                     <ck>
10562 :gosub 10400                     <kl>
10563 :gosub 10100                     <bp>
10564 :gosub 10420                     <km>
10565 :gosub 10100                     <bm>
10566 :gosub 10440                     <lf>
10567 :gosub 10100                     <cb>
10568 :gosub 10460                     <lk>
10569 :gosub 10100                     <cc>
10570 iw= 30                           <ak>
10571 c= 3                             <ed>
10572 :gosub 10480                     <lm>
10573 return                           <hk>
10580 :gosub 10500                     <ke>
10581 :gosub 10520                     <ci>
10582 :gosub 10540                     <lb>
10583 :gosub 10560                     <ge>
10584 return                           <jb>
10600 zl= 10                           <il>
10601 :do until c> 100                 <pe>
10602 :gosub 10000                     <mp>
10603 c=c+zl                           <gi>
10604 loop:c=0                         <ob>
10605 return                           <ll>
10620 zl= 10                           <bj>
10621 :do until c> 60                  <jc>
10622 :gosub 10300                     <pn>
10623 :gosub 10320                     <ia>
10624 c=c+zl                           <ll>
10625 loop:c=0                         <ig>
10626 return                           <of>
10641 c= 1                             <hh>
10642 :gosub 10480                     <bd>
10643 c= 10                            <dp>
10644 :gosub 10360                     <ai>
10645 return                           <al>
10646 rem demo.pkt==============p4     <gf>
10647 rem nur in verbindung            <mb>
10648 rem mit igel-programm            <pi>
10649 rem                              <be>
10650 rem achtung !!!                  <od>
10651 rem programm darf nicht          <ig>
10652 rem umnummeriert werden.         <ih>
10653 rem ======================       <hl>
```

WANDELN VON SEQUENTIELLEN DATEIEN IN PROGRAMMFILES

Watson und Holmes zeigen ihren Spürsinn

Zum Wandeln von sequentiellen Files in Programmfiles bedarf es keines Programmes. Der Rechner ist dazu bereits selbst in der Lage. Komfortables Mergen und eine neue Art zu programmieren werden möglich.

In London, Bakerstreet Nummer 27, ereignete sich eine denkwürdige Begebenheit, die für alle Besitzer eines Commodore-Heim-Computers von Nutzen sein dürfte. An besagtem Orte befindet sich nicht etwa eine Soft- oder Hardware-Firma. Auch kein Computerspezialist hat hier seinen Wohnsitz. Wohl aber wohnt dort jemand, der wegen seines Spürsinnes in besonders verzwickten Fällen Weltruhm erlangte.

Er ist Junggeselle, doch ist er nicht gerade einsam. Denn einerseits hat er seine Arbeit, die ihn oft bis zur Erschöpfung in Anspruch nimmt, andererseits teilt er das Haus mit einem Freund, einem vorzeitig wegen einer Kriegsverletzung pensionierten Militärarzt. Sie leben in getrennten Räumen, nur das Wohnzimmer teilen sie sich. Da ihre Charaktere sich gegenseitig ergänzen und beide zum Ausgleich neigen, gestaltet sich ihr Zusammenleben ziemlich unproblematisch.

Dr. Watson, der frühere Militärarzt, braucht sich über Langeweile nicht zu beklagen. Als guter Freund und ständiger Begleiter steht er Holmes, dem berühmten Privatdetektiv, in all seinen Fällen zur Seite und ist stets um dessen Wohl und Gesundheit besorgt. Er entlastet Holmes durch seine Assistenz und achtet darauf, daß sein Freund sich nicht total übernimmt. Öfters schon mußte er ihn gemahnen, unbedingt zur Regeneration der Kräfte einen Erholungsurlaub einzulegen.

Da Rauchen der Gesundheit abträglich ist, hält er davon nichts und hätte Holmes nur allzugerne davon abgebracht. Er hatte damit allerdings nicht viel Erfolg, obwohl Holmes aus Rücksicht auf manches Pfeiflein verzichtete. Beim Nachdenken über schwer durchschaubare Sachverhalte verschafft ihm die durch die kräuselnden Rauchwölkchen erzeugte Atmosphäre eine Entspannung und geistige Klarheit, wie sie andere nur durch viele Jahre praktizierte Meditationsübungen zu erreichen vermögen.

Ein weiterer Punkt findet ebenso wenig Watsons uneingeschränkte Zustimmung. Er, der einen fast pedantisch zu nennenden Hang zur Ordnung hat, stört sich manchmal doch etwas daran, wenn es Holmes damit nicht so genau nimmt.

Holmes hatte erst am vorigen Tag einen aufsehenerregenden Kriminalfall abgeschlossen, worin ihn die Polizei um seine Mitarbeit gebeten hatte, da sie mit ihren Ermittlungen nicht weiterkam. Etliche Tage

und Nächte war er unterwegs gewesen und entging einmal nur knapp einem auf ihn verübten Anschlag. Watson als Freund und Arzt hatte ihn dringend ermahnt, zumindest für einen Tag auszuruhen und dabei jeden Gedanken an Mord und Verbrechen zu verwerfen.
Doch es sollte wieder einmal anders kommen. Zwar ging es nicht um Verbrechen, jedoch trotzdem um einen Fall, der Holmes' ganzen kriminalistischen Spürsinn erforderte.
Als Holmes, der erst gegen Mittag erwacht war, die Treppe zum Wohnzimmer herunterkam, verschlug es ihm fast den Atem. Nein, so etwas hatte er wirklich nicht erwartet, zumindest nicht von Watson mit seinem Ordnungssinn. Der Schreibtisch war völlig übersät von Disketten. Übers ganze Wohnzimmer verstreut lagen Computerzeitschriften, teilweise aufgeschlagen. Watson schob eifrig eine Diskette nach der anderen in das Diskettenlaufwerk und warf sie kopfschüttelnd auf den immer größer werdenden Haufen. Den in der Türe stehenden Holmes schien er nicht zu bemerken.

WANDLUNG PER PROGRAMM

Plötzlich sprang Watson auf, eilte auf den Wohnzimmerschrank zu, riß einen weiteren Packen Computerzeitschriften heraus und blätterte diese wie wild durch, während er durch das Zimmer tigerte. Urplötzlich stoppte er, als er Holmes stehen sah, der ihn verwundert musterte.
Holmes fragte ihn nach dem Grund des seltsamen Gebarens. Watson bot ihm den noch frei gebliebenen Sessel an und schilderte, was vorgefallen war: Einer seiner Bekannten aus dem Computerclub sei vorbeigekommen und habe ihn um seine Mithilfe bei einem Problem gebeten. Dieser Bekannte hätte sich aus einer Mailbox ein BASIC-Programm geholt und es auf Diskette abgespeichert. Leider liege es nun als sequentielles File vor, das zwar mit einer Textverarbeitung begutachtet werden könne, aber so natürlich nicht lauffähig sei.
Er, Watson, habe ihm Hilfe versprochen, da er sich noch an ein Programm erinnern konnte, das sequentielle Files in Programm-Files verwandelte. Er habe es leider weder auf seinen Disketten noch in den bis jetzt durchgeschauten Computerzeitschriften entdecken können.
Holmes sah ihn lange mit nachdenklichem Ausdruck an, ehe er sich schließlich zu diesem Fall äußerte. „Wenn das Umwandlungsprogramm nicht zu entdecken ist, so gibt es noch die Möglichkeit des Selbermachens", sagte er. „Es wäre nur zu klären, wie dabei vorzugehen ist. Watson, was können Sie mir darüber sagen? Ist Ihnen vielleicht noch ein Anhaltspunkt in Erinnerung geblieben?"
„Das sequentielle File wurde vom Datenträger in den Hauptspeicher des Rechners geladen, was nicht besonders schwer zu realisieren sein dürfte. Neben dieser Laderoutine ist aber noch eine zweite Routine erforderlich, die den im Hauptspeicher abgelegten Text zeichenweise in den Tastaturpuffer schiebt", antwortete Watson.
Es dauerte ein wenig, bis Holmes sich die Sache durch den Kopf hatte gehen lassen. „Ja, so wäre es durchaus durchführbar. Eine in den Systeminterrupt eingebundene Maschinenroutine könnte alle fünfzigstel Sekunden ein Zeichen in den Tastaturpuffer senden, sofern dieser nicht bereits voll ist, was aber leicht abzufragen wäre. Nur erscheint mir dieses Verfahren doch ein wenig aufwendig.

EINGABEGERÄT UMDEFINIEREN

Watson, können Sie sich vorstellen, daß die ganze Sache auch viel einfacher zu bewerkstelligen wäre?"
Watson bemerkte sophistisch: „Daß es geht, darüber habe ich keine Zweifel, wie es geht, kann ich mir leider nicht denken." „Lieber Watson, worin besteht für den Rechner der Unterschied zwischen Tastatur und Diskette?", hakte Holmes nach. „Die Tastatur ist das Gerät, über das der Rechner sowohl direkte Kommandos als auch zu erfassende BASIC-Zeilen annimmt. Von der Diskette hingegen funktioniert das wohl nicht", antwortete Watson unsicher.
Schließlich brachte es Holmes auf den Punkt: „Nimmt der Rechner prinzipiell keine Kommandos von der Diskette entgegen oder nur, weil er nicht weiß, daß er sie sich von dort herunterlesen soll?"
Watson geriet ins Schwanken. Er mußte notgedrungen zugeben: „Wenn Disketten- oder Kassettenlaufwerk als Eingabegerät definiert wären, so bestünde die Möglichkeit, daß der Rechner seine Direktbefehle und BASIC-Zeilen sich von dort holt. Ob dieses zutrifft, kann jedoch nur ein Test bestätigen."
Holmes, dem anzumerken war, daß er Watsons Gedankengänge bereits vorausgesehen hatte, bestätigte das Gesagte und fing an, ihm zu diktieren, was er in seinen Rechner eingeben sollte:

```
1 OPEN8,8,8,"TEST,
  S,W"
2 CMD8
3 ?"GRAPHIC1,1"
4 ?"CIRCLE,100,100,
  50"
5 PRINT#8,"SYS
  65484"
6 CLOSE8
```

Nach dem Start mit RUN würde eine Datei namens TEST auf der Diskette eingerichtet, die einen GRAPHIC-, CIRCLE- und SYS-Befehl jeweils in ASCII-Darstellung enthalte, erklärte Holmes. Die Datasettenbenutzer hätten die ersten zwei Zeilen davon zu ersetzen gegen:

```
1 OPEN1,1,1,"TEST"
2 CMD1
```

EINGABEKANAL UMSCHALTEN

Auf die Frage, was der SYS-Befehl am Ende bedeute, antwortete Holmes: „Es handelt sich hierbei um die Kernal-Routine CLRCHN, die die Eingabe wieder auf die Tastatur umlegt. Genausogut wäre auch ein Wort gewesen, das der Rechner nicht ver-

steht. Würden wir den Namen HOLMES statt SYS65484 schreiben, so würde der Rechner infolge der Fehlerbehandlung ebenfalls auf die Tastatur umschalten. Störend wäre in unserem speziellen Beispiel allerdings gewesen, daß dadurch wieder auf den Textbildschirm umgeschaltet werden würde und die Graphic somit nicht lange sichtbar bliebe. Wollen wir doch gleich einmal die Direkteingabe von externem Gerät ausprobieren."

```
OPEN8,8,8,"TEST,S,R"
POKE2035,8
SYS65478
```

Bei Datasette ist die OPEN-Anweisung zu ersetzen:

```
OPEN8,1,0,"TEST"
```

Am Ende sollte der Eingabekanal wieder mit CLOSE8 abgeschlossen werden.

Watson erinnerte sich des Maschinensprachkurses in der C16/P4-SPECIAL Nr. 2 und identifizierte den SYS-Aufruf: „Bei SYS65478 handelte es sich um CHKIN, wodurch ein Eingabekanal geöffnet wird. Vorher ist dem X-Register die Kanalnummer zu übergeben, was in BASIC durch Poken an die Adresse 2035 geschieht." Er fing an zu strahlen, denn das Problem schien bereits gelöst. Mit einem OPEN-, einem POKE-, und einem SYS-Befehl ließe sich sein sequentiell vorliegendes Programm in ein normales BASIC-Listing umwandeln. Gleich machte er sich ans Werk. Als er das Programm auflistete, stutzte er: „Nur eine einzige Programmzeile wurde übernommen." Um einen Bedienungsfehler auszuschließen, probierte er es noch ein paarmal, jedoch immer wieder mit dem gleichen Ergebnis.

Holmes hatte sich weit in seinem Sessel zurückgelehnt und paffte aus seiner Pfeife Rauchringe und sich kräuselnde Wölkchen. Watson, der ihn gut kannte, sah an seinem Ausdruck, daß er anscheinend eine Spur gefunden hatte und er ihn jetzt keineswegs in seinen Überlegungen stören durfte. Nach einiger Zeit hörte er: „Watson, ich hab es. Um sicher zu sein, brauche ich allerdings nur kurz das ROM-Listing."
Watson holte es aus dem Bücherregal und reichte es ihm gespannt. Nach kurzem Blättern sagte Holmes: „Genau so, wie ich es mir gedacht hatte. Bei der Übernahme einer BASIC-Zeile wird die CLR-Routine aufgerufen. Watson, sicherlich wissen Sie, was der BASIC-Befehl CLR tut."
Watson wußte es: „CLR löscht alle Variablen."
„Das ist aber nicht alles, Watson. CLR schließt auch die I/O-Kanäle und beseitigt zudem alle Einträge in der Filetabelle."
„Schade", bemerkte Watson enttäuscht. „Damit ist die Sache wohl gestorben."

VERBIEGEN DES VEKTORS ICLALL

„Mitnichten, lieber Watson. Eigentlich ist es ja nicht die CLR-Routine, die das Schließen des Kanals vornimmt, sondern eine Routine, die von CLR aufgerufen wird, nämlich die Kernal-Routine CLALL." „Die steht an Adresse $FFE7, dezimal 65511, im ROM, was uns wenig nützt. Jedoch habe ich erst vor kurzem etwas über Verbiegen von Sprungvektoren gelesen. CLALL müßte indirekt über den Sprungvektor ICLALL in den Adressen $032A und $032B verzweigen."
Anerkennend sagte Holmes: „Gut, Watson, damit haben wir die Lösung. Wir schalten CLALL aus, indem wir ICLALL auf einen Rücksprungcode RTS zeigen lassen. Ein solcher findet sich genau vor der CLALL-Routine. Wir brauchen somit nur den Inhalt der Speicherzelle $032A, dezimal 810, um eins erniedrigen. Die Methode, um sequentiell vorliegende Programme in BASIC-Programme umzuwandeln, ist gefunden."

```
OPEN8,8,8,"SEQFILE
 ,S,R"
POKE2035,8
SYS65478
```

FEHLENDE TASTATURUMSCHALTUNG

Wenn dem zu wandelnden sequentiellen File am Schluß der SYS-Befehl oder eine sonstige Ende-Kennung fehlt, so ist das Ende der Bearbeitung daran zu erkennen, daß das rote Floppy-Lämpchen zwar weiter brennt, die Floppy aber offensichtlich nichts mehr tut. Mit Run/Stop- und Reset-Taste läßt sich über den Sprung in den Monitor das Umschalten auf die Tastatur erzwingen. Ein CLOSE8 sollte am Schluß nicht fehlen, wenn es vielleicht beim Monitorbreak auch unnötig ist. Besser zuviel als zuwenig.
Watson war begeistert von den Möglichkeiten dieses Verfahrens. Er benutzt es jetzt, um Programme mit dem Textverarbeitungsprogramm SCRIPT/PLUS zu schreiben, zum Mergen und für längere Grafik-Programme, für die normalerweise der Hauptspeicher des C16 nicht ausreicht.

ZUSAMMENBINDEN VON PROGRAMMEN

Komfortables Mergen erlaubt das Turbo/Plus-Modul von Kingsoft. Bisherige Tips und Tricks in Computerzeitschriften beschrieben nur das Zusammenbinden aufeinanderfolgender Programmteile. Mit der Umwandlungsmethode kann auf einfache Weise wie bei Turbo/Plus ein Einfügen von Programmteilen bewerkstelligt werden, da das Einbinden der Rechner selbständig besorgt. Es muß nur vorher ein Programm oder Programmteil sequentiell gewandelt worden sein. Dies geschieht durch:

```
OPEN8,8,8,"SEQFILE,
  S,W":CMD8:LIST:
  PRINT#8"SYS
  65484":CLOSE8
```

Mit dem Listbefehl können auch einzelne Abschnitte ausgewählt werden, zum Beispiel: LIST 100–200.

NEUE PROGRAMMIERMÖGLICHKEITEN

Eine neue Art der Programmierung wird durch Direktbefehle vom Datenträger möglich. Besonders interessant ist die Mischung von BASIC-Eingabezeilen und Direktbefehlen. BASIC-Eingabezeilen können gelesen und gestartet werden. Nach dem Programmende eines solchen Teiles kann dieses durch Direktbefehle ganz oder teilweise gelöscht werden. Nach dem anschließenden Dazuladen neuer Zeilen können Sie abermals starten, und so weiter. Lassen Sie Ihre Fantasie spielen, und Sie werden sicherlich eine Menge Anwendungsmöglichkeiten finden.
Übrigens beschränkt sich diese Methode nicht nur auf BASIC-Listings. Sie ist auch in Verbindung mit dem Maschinenmonitor anwendbar. Sowohl Hexdumps wie Mnemonics lassen sich in Objektcode wandeln.
Wir verabschieden uns jetzt von Watson und Holmes und sind gespannt, welche Einsichten die beiden uns das nächste Mal bringen werden.

a.m. □

STÜTZPUNKT-K

Retten Sie Ihre Kameraden

Mit einem Panzerspähwagen müssen Sie feindliche Flugzeuge, die es auf den Munitionstransport abgesehen haben, abwehren.

In diesem Spiel müssen Sie einen Militär-Stützpunkt bewachen. Ihnen steht ein Panzerspähwagen mit aufgebautem Maschinengewehr zur Verfügung. Nach dem Spielstart sehen Sie im unteren Bildschirmbereich sechs LKWs, mit Munition beladen. Vor den Bergen, in der Mitte, stehen Sie mit dem Panzerspähwagen. Gesteuert wird mit Joystick Port 1. Sie werden aufgefordert, zum Spiel den Feuerknopf zu drücken. Der Stützpunkt wird jetzt von feindlichen Flugzeugen angegriffen. Sie können den Panzerspähwagen nach rechts oder links steuern. Versuchen Sie, die Flugzeuge durch Druck auf den Feuerknopf abzuschießen, sie haben es nämlich auf die LKWs abgesehen. Drei Level müssen überstanden werden. In jedem greifen zwölf Flugzeuge an. Ihre Geschwindigkeit steigert sich in jedem Level.
Haben die Flugzeuge alle sechs LKWs abgeschossen, ist das Spiel zu Ende. Überstehen Sie alle drei Level und ist mindestens noch 1 LKW verschont geblieben, wird Ihnen angezeigt, wie viele Flugzeuge Sie vernichtet haben und wie viele LKWs Sie retten konnten. □

```
10 rem stuetzpunkt k ==========c16 <en>
20 rem (p) commodore welt  team    <ho>
30 rem ========================== <ng>
40 rem (c) j.kienberger            <ai>
50 rem                             <pd>
60 rem                             <ah>
70 rem basic v3.5                  <nl>
80 rem c16/116/plus4               <ki>
90 rem ========================== <jg>
100 poke55,255:poke56,47:clr       <pe>
110 trap1620                       <da>
120 rem zeichensatz normal         <ai>
130 key1,"p065298,198:p065299,208:
p065287,8,"+chr$(13)               <no>
140 gosub1660:color0,1:scnclr      <hd>
150 char,13,11,wh$+"bitte warten"  <fi>
160 poke65298,peek(65298)and251    <fo>
170 poke65299,peek(65299)and3or48  <il>
180 fort=0to17:reada               <ie>
190 poke1630+t,a:next:sys1630      <id>
200 fork=12800to13048step8         <pc>
210 fort=0to7:reada                <bn>
220 ifa=-1then570                  <gi>
230 pokek+t,a:next:next            <mb>
240 data162,0,189,0,208,157,0,48,1
89,0,209,157,0,49,202,208,241,96   <md>
250 data0,0,0,0,15,255,34,0        <nj>
260 data16,16,254,56,255,255,254,1
24                                 <fm>
270 data0,0,0,0,224,254,136,0      <bl>
280 data1,1,33,29,31,27,13,254     <ba>
290 data128,128,132,184,248,216,17
6,127                              <kh>
300 data254,13,27,31,29,33,1,1     <ig>
310 data127,176,216,248,184,132,12
8,128                              <dk>
320 data0,0,0,0,0,1,63,127         <fp>
330 data0,30,62,126,194,130,130,25
5                                  <mm>
340 data0,255,255,255,255,255,255,
255                                <ai>
350 data127,127,96,78,95,95,31,14  <fm>
360 data255,255,255,126,126,126,0,
0                                  <fd>
370 data255,255,255,254,254,254,0,
0                                  <gp>
380 data255,255,7,115,251,251,248,
112                                <gm>
390 data0,0,0,0,2,6,14,31          <hk>
400 data32,46,46,46,36,46,126,255  <fk>
410 data0,0,0,0,64,96,112,248      <gl>
420 data63,127,225,221,34,42,34,28 <jl>
430 data255,255,255,255,255,255,0,
0                                  <kc>
440 data252,254,135,187,68,84,68,5
6                                  <im>
450 data1,3,7,15,31,63,127,255     <jm>
460 data128,192,224,240,248,252,25
4,255                              <if>
470 data255,255,255,255,255,255,25
5,255                              <bl>
480 data0,0,0,3,3,0,0,15           <mj>
490 data24,24,24,255,255,126,255,2
55                                 <lf>
500 data0,0,0,192,192,0,0,240      <lf>
510 data1,127,127,24,24,0,0,0      <op>
520 data255,255,255,51,49,0,0,0    <hj>
530 data255,255,255,255,255,255,12
6,0                                <mc>
540 data255,255,255,204,140,0,0,0  <cj>
550 data128,254,254,24,24,0,0,0    <gc>
560 data-1                         <ck>
570 a$=bk$+chr$(96)+"AB":a1$=b3$+b
3$+b3$:a4$=wh$+"CD"+b2$+"EF"       <fk>
580 a2$=bk$+b3$+"WXY"+b3$+b2$+c1$+
c1$+c1$+"Z"+ym$+yn$+yo$+ys$        <lo>
590 a5$=b3$+b3$+b2$+b3$+b3$        <co>
600 b1$=b3$+b3$+b3$+b3$+b3$+b2$+c1
$+c1$+c1$+b3$+b3$+b3$+b3$+b3$      <dn>
610 b$=bk$+"NOP"+b2$+c1$+lb$+"QRS" <fk>
620 a6$=ye$+"CD"+b2$+"EF":n1$=b3$+
b4$+b3$+b4$+b3$+b4$+b3$            <ch>
```

```
630 c$=re$+"GH"+bk$+"II"+b2$+c1$+c
1$+re$+"JK"+bk$+"LM"                 <gp>
640 e$=b3$+b3$+b3$+b3$+b2$+c1$+c1$
+b3$+b3$+b3$+b3$                     <jn>
650 z$=gr$+"EF":n$=wh$+chr$(46)+b4
$+chr$(46)+b4$+chr$(46)+b4$+chr$(4
6)                                   <fk>
660 rem ---------------------        <aj>
670 rem -- bildschirmaufbau--        <bg>
680 rem ---------------------        <pg>
690 scnclr:vol8:color0,2,5:color4,
10,1:poke65286,11                    <ho>
700 char,26,7,wh$+"TU":char,38,7,"
TU":char,0,8,br$+"TU":char,8,8,"TU
":char,25,8,"TVVU"                   <pg>
710 char,37,8,"TVV":char,0,9,"VVU"
:char,7,9,"TVVU":char,14,9,"TU"      <km>
720 char,19,9,"TU":char,24,9,"TVVV
VU":char,31,9,"TU":char,36,9,"TVVV
"                                    <ki>
730 char,0,10,"VVVU":char,6,10,"TV
VVVU":char,13,10,"TVVU":char,18,10
,"TVVU"                              <nh>
740 char,23,10,"TVVVVVVVVVU":char,
35,10,"TVVVV":char,0,11,"VVVVU"      <cb>
750 char,5,11,"TVVVVVVVVVVVVVVVVVV
VVVVVVVVVVVVVVV"                     <pn>
760 fort=0to39:char,t,12,"V":next    <pm>
770 fort=0to39:char,t,13,"V":next    <ei>
780 char,0,22,z$:char,2,20,z$:char
,38,21,z$:char,36,19,z$              <hn>
790 fort=4to34step3:char,t,18,z$:n
ext                                  <ao>
800 fort=3to33step6:char,t,21,c$:n
ext                                  <hc>
810 l=0:char,17,24,rn$+"level"+rf$
+str' l):poke65286,27                <nd>
820 rem --------------------         <lk>
830 rem -- hauptprogramm  --         <mn>
840 rem --------------------         <pa>
850 x=16:y=15:s=2:m=2:u=2:w=2:z=2:
v=2                                  <gm>
860 char,x,y,b$:char,5,4,bk$+"zum
spiel feuerknopf druecken"           <ak>
870 ifjoy(1)=128then890              <pp>
880 goto870                          <pg>
890 tt=1600:aut=6:su=0:char,5,4,"
                               ":rem
29 spaces                            <ag>
900 l=l+1:tr=0                       <pd>
910 ifl=2thencolor0,9,6:tt=1100      <ke>
920 ifl=3thencolor0,7,6:tt=650       <pj>
930 ifl>3then1430                    <og>
940 ifaut<1then1540                  <np>
950 k=int(rnd(0)*6)+1:d=tt:tr=tr+1   <jc>
960 onkgoto970,980,990,1000,1010,1
020                                  <ac>
970 k=4:goto1030                     <en>
980 k=10:goto1030                    <am>
990 k=16:goto1030                    <kf>
1000 k=22:goto1030                   <lm>
1010 k=28:goto1030                   <lf>
1020 k=34:goto1030                   <kn>
1030 k1=int(rnd(0)*5)+1              <mo>
1040 iftr>12then1360                 <bj>
1050 ti$="000000":wait165,64         <ei>
1060 char,k,k1,a$                    <cd>
1070 char,22,24,bk$+str$(l)          <ke>
1080 d=d-50                          <bn>
1090 ifd<450thenchar,k,k1,a2$        <mp>
1100 ifd=0thenchar,k+2,k1+1,wh$+ch
r$(42):goto1250                      <bc>
1110 ifjoy(1)=3thengosub1150         <ep>
1120 ifjoy(1)=7thengosub1180         <ag>
1130 ifjoy(1)=128thengosub1210       <cp>
1140 goto1080                        <hi>
1150 x=x+6:ifx>34thenx=34            <ok>
1160 char,x-6,y,b1$:char,x,y,b$      <aj>
1170 return                          <ll>
1180 x=x-6:ifx<4thenx=4              <gh>
1190 char,x+6,y,b1$:char,x,y,b$      <md>
1200 return                          <ph>
1210 char,x+1,k1,n$                  <kl>
1220 ifx+1=k+1then1330               <na>
1230 char,x+1,k1,n1$                 <eo>
1240 return                          <eh>
1250 ifk=4thens=s-1:ifs<1thengosub
1630:goto910                         <kd>
1260 ifk=10thenm=m-1:ifm<1thengosu
b1630:goto910                        <ea>
1270 ifk=16thenu=u-1:ifu<1thengosu
b1630:goto910                        <fn>
1280 ifk=22thenv=v-1:ifv<1thengosu
b1630:goto910                        <mi>
1290 ifk=28thenw=w-1:ifw<1thengosu
b1630:goto910                        <nf>
1300 ifk=34thenz=z-1:ifz<1thengosu
b1630:goto910                        <jj>
1310 char,k,21,a6$:sound3,900,7:so
und3,500,7:sound3,100,7:sound1,150
,4                                   <ci>
1320 char,k-1,21,e$:aut=aut-1:char
,k,k1,b1$:goto910                    <jk>
1330 char,k+1,k1,a4$:sound3,850,6:
sound3,500,6:sound3,50,7             <lh>
1340 char,x+1,k1,n1$                 <dn>
1350 char,k+1,k1,a5$:su=su+1:char,
k,k1,b1$:goto910                     <lh>
1360 ifaut<1then1540                 <ak>
1370 restore1640                     <jp>
1380 readi,j                         <ja>
1390 ifi=-1then1420                  <he>
1400 sound1,i,j                      <nc>
1410 goto1380                        <ho>
1420 char,k,k1,b1$:goto900           <cf>
1430 rem ---------------             <lm>
```

```
1440 rem --  ende  --                <oi>
1450 rem ---------------             <ga>
1460 restore1650                     <pe>
1470 reado,p                         <fe>
1480 ifo=-1then1510                  <aa>
1490 sound1,o,p                      <il>
1500 goto1470                        <ln>
1510 color1,1:zz=aut:scnclr:char,6
,11,"":print"sie konnten "zz" lkw
retten"                              <kk>
1520 aa=su:char,1,15,"":print"sie
haben "aa" flugzeuge abgeschossen"   <hc>
1530 poke239,0:end                   <ng>
1540 color1,1:fort=900to20step-120
:sound2,t,6:next:scnclr              <no>
1550 char,6,4,"sie haben keinen lk
w mehr"                              <gb>
1560 char,7,8,"neues spiel= joy fe
uer"                                 <ej>
1570 char,7,14,"ende = joystick re
chts"                                <lk>
1580 poke239,0                       <jj>
1590 ifjoy(1)=128thenclr:gosub1660
:goto570                             <kh>
1600 ifjoy(1)=3thenpoke239,0:end     <oj>
1610 goto1590                        <jd>
1620 scnclr:poke65298,199:poke6529
9,208:printchr$(14)err$(er)"error
in "el:help:end                      <lm>
1630 char,k,k1,b1$:return            <fk>
1640 data834,7,854,7,864,7,881,7,8
97,7,911,7,897,7,911,7,864,7,854,7
,-1,0,0                              <mh>
1650 data834,7,854,7,864,7,864,7,8
97,7,881,7,881,7,881,7,897,7,-1,0,
0                                    <ba>
1660 rem nachspann ===============   <ag>
1670 rem zeichensatz/graphik *       <bo>
1680 wh$=chr$(005):c4$=chr$(017)     <hj>
1690 rn$=chr$(018):re$=chr$(028)     <nk>
1700 gr$=chr$(030):bk$=chr$(144)     <fi>
1710 rf$=chr$(146):br$=chr$(149)     <mf>
1720 lb$=chr$(154):c1$=chr$(157)     <cj>
1730 ye$=chr$(158):ym$=chr$(219)     <cn>
1740 yn$=chr$(220):yo$=chr$(221)     <oj>
1750 ys$=chr$(255)                   <pk>
1760 b2$=b2$+c4$+c1$+c1$             <mm>
1770 b3$=chr$(32):b4$=b4$+c4$+c1$    <mp>
1780 return                          <id>
1790 rem ========================    <kl>
1800 rem 12277 bytes memory          <ck>
1810 rem 05302 bytes program         <if>
1820 rem 00343 bytes variables       <ic>
1830 rem 00000 bytes arrays          <gp>
1840 rem 00518 bytes strings         <ba>
1850 rem 04096 bytes zeichensatz     <oe>
1860 rem 02018 bytes free (0)        <og>
1870 rem ========================    <oc>
```

KATER TOM

Tom und Jerry-Show

Nicht auf Können, sondern auf Glück kommt es bei der Mäusejagd an. Zu setzen sind allerdings keine Chips wie beim Roulette, sondern Mausefallen. Den dritten Level schaffen kann nur der, dem Fortuna hold ist und bei dem die Mäuse aus den richtigen Löchern kommen.

In diesem Spiel muß Kater Tom die Mäuse mit der Mausefalle fangen. Kater Tom kann mit den Cursortasten hoch und runter gesteuert werden. Durch Drücken der Space-Taste wird eine Mausefalle gesetzt. Haben Sie drei Fallen gesetzt, kommen die Mäuse aus verschiedenen Löchern.
Das Spiel hat drei Level. Jeder Level erstreckt sich auf fünf Runden. Wollen Sie Level 2 erreichen, müssen Sie mindestens 80 Punkte schaffen.
Für jede Maus, die in die Falle läuft, erhalten Sie zehn Punkte.
In Level 1 kommen fünf Mäuse aus ihren Löchern. Bei Level 2 sind es nur noch vier Mäuse und bei Level 3 erscheinen nur noch drei der Nager.
Haben Sie Level 2 erreicht, so brauchen Sie mindestens 160 Punkte, um in Level 3 zu kommen.
Gewonnen ist das Spiel, wenn in Level 3 mindestens 240 Punkte erreicht worden sind.
Das Spiel erklärt sich weitgehend selbst.
Nach der Anleitung und Titelmusik dauert es einen Moment, bis der Bildschirm aufgebaut ist. □

```
10 rem kater tom===============p4    <jm>
20 rem (p) commodore welt      ==    <hf>
30 rem ========================      <mm>
40 rem (c) by j. kienberger    ==    <mm>
50 rem                         ==    <if>
60 rem                         ==    <nd>
70 rem basic 3.5               ==    <mg>
80 rem plus4 (c16/116 + 64 kb) ==    <ae>
90 rem ========================      <km>
100 poke55,255:poke56,47:clr         <pe>
110 gosub1710:poke65286,11           <ep>
120 poke65298,peek(65298)and251      <pj>
130 poke65299,peek(65299)and3or48    <ao>
140 fori=0to17:reada                 <pl>
150 poke819+i,a:next:sys819          <ii>
160 fork=12808to12952step8           <pc>
170 fori=0to7:reada                  <bk>
180 ifa=-1then400                    <ke>
190 pokek+i,a:nexti,k                <ke>
200 data162,0,189,0,208,157,0,48,1
89,0,209,157,0,49,202,208,241,96     <kn>
```

```
210 data0,0,0,0,0,8,28,62                <kd>
220 data0,0,0,0,0,0,63,255               <eo>
230 data0,0,0,0,0,6,143,255              <md>
240 data0,0,0,0,0,48,248,248             <df>
250 data103,199,7,7,7,7,7,7              <ei>
260 data255,255,255,255,253,254,25
4,254                                    <if>
270 data223,185,185,191,190,215,21
6,239                                    <hi>
280 data252,158,158,254,126,236,28
,248                                     <jd>
290 data3,3,1,0,0,0,0,0                  <gb>
300 data253,251,255,0,0,0,0,0            <jb>
310 data247,251,255,6,0,0,0,0            <bj>
320 data240,224,192,192,0,0,0,0          <pp>
330 data0,1,7,28,240,192,255,255         <ll>
340 data112,192,0,0,112,112,254,25
5                                        <cm>
350 data0,4,15,31,55,127,255,10          <no>
360 data0,0,240,250,252,248,240,48       <pl>
370 data0,1,7,29,243,199,255,255         <oi>
380 data112,192,193,255,126,254,25
4,255                                    <ag>
390 data-1                               <nf>
400 a$=oe$+"ABCD"+w$+"EFGH"+w$+"IJ
KL":b$="     "+w$+"     "+w$+"     "    <dg>
410 c$=wh$+"MN":d$=wh$+"QR":e$=wh$
+"OP":f$="  ":poke65286,27               <lj>
420 gosub1360: scnclr:color0,1,1:p
oke65286,11                              <em>
430 rem ========================         <kh>
440 rem      bildschirmaufbau            <lh>
450 rem ========================         <ia>
460 char,3,1,ye$+"punkte":char,27,
1,ye$+"runde":char,16,1,ye$+"level
"                                        <ij>
470 x=20:y=3:r=1:p=0:mm=5:l=1:vol8       <ak>
480 x=x+1:ifx=38theny=y+2:x=21           <pa>
490 ify>21then520                        <nc>
500 char,x,y,gr$+"G"                     <al>
510 goto480                              <lf>
520 x=2:y=3                              <ec>
530 x=x+1:ifx>8theny=y+2:x=3             <kg>
540 ify>21then570                        <kg>
550 char,x,y,gr$+"G"                     <ng>
560 goto530                              <ng>
570 fort=3to21:char,38,t,gr$+"G":c
har,2,t,gr$+"G":next:poke65286,27        <bk>
580 char,32,1,wh$+str$(r):fort=2to
21:char,10,t,"         ":next:rem
9 spaces                                 <id>
590 char,22,1,wh$+str$(l)                <pg>
600 char,b+1,a(i),f$:x=12:y=11:e=1
:char,x,y,a$:char,15,22,b$               <dn>
610 char,3,a(1),f$:char,3,a(2),f$:
char,3,a(3),f$:char,3,a(4),f$:char
,3,a(5),f$                               <dp>
620 poke1344,64                          <fi>
630 getkeys$                             <dl>
640 ifs$=chr$(32)then:e=e+1:gosub7
40                                       <pa>
650 ifs$=chr$(17)thengosub710            <aa>
660 ifs$=chr$(145)thengosub680           <oi>
670 goto630                              <mf>
680 y=y-2:ify<3theny=3                   <aa>
690 char,x,y+3,b$:char,x,y,a$:soun
d1,300,1                                 <la>
700 return                               <al>
710 y=y+2:ify>19theny=19                 <hc>
720 char,x,y-3,b$:char,x,y,a$:soun
d1,300,1                                 <oo>
730 return                               <eh>
740 char,x+5,y+1,c$                      <cg>
750 ife=2thenx1=x+5:y1=y+1               <eb>
760 ife=3thenx2=x+5:y2=y+1               <ga>
770 ife=4thenx3=x+5:y3=y+1               <aj>
780 ife=3andy2=y1thene=e-1:goto630       <mo>
790 ife=4andy3=y2ory3=y1thene=e-1:
goto630                                  <ga>
800 ife>3then820                         <cn>
810 return                               <oi>
820 char,x,y,b$:char,15,22,a$            <cf>
830 fori=1tomm                           <ae>
840 a=int(rnd(0)*9)+1                    <je>
850 ifa=1thenk=4                         <cp>
860 ifa=2thenk=6                         <eg>
870 ifa=3thenk=8                         <go>
880 ifa=4thenk=10                        <mk>
890 ifa=5thenk=12                        <fj>
900 ifa=6thenk=14                        <io>
910 ifa=7thenk=16                        <pg>
920 ifa=8thenk=18                        <pm>
930 ifa=9thenk=20                        <ck>
940 forkk=1toi                           <bg>
950 ifk=a(kk)then840                     <fp>
960 nextkk                               <af>
970 a(i)=k:b=35                          <jc>
980 b=b-1:ifb<3thengoto1040              <ob>
990 ifb=x1anda(i)=y1thengosub1110:
goto1040                                 <pk>
1000 ifb=x2anda(i)=y2thengosub1110
:goto1040                                <lo>
1010 ifb=x3anda(i)=y3thengosub1110
:goto1040                                <ii>
1020 char,b,k,e$:char,b+2,k,f$:sou
nd1,900,1:fort=1to15:nextt               <nc>
1030 goto980                             <ha>
1040 nexti                               <jo>
1050 r=r+1:ifr>5then1190                 <hp>
1060 char,15,22,b$:char,3,k(1),f$:
char,3,k(2),f$:char,3,k(3),f$            <af>
1070 char,3,k(4),f$:char,3,k(5),f$
:goto580                                 <lm>
1080 rem ======================          <mf>
1090 rem  maus in der falle              <fe>
1100 rem ======================          <fl>
```

```
1110 sound3,600,10:sound1,200,5:ch
ar,b,a(i),d$                        <od>
1120 fort=1to900:next               <am>
1130 char,15,22,b$:char,x+5,k-1,a$  <cd>
1140 fort=1to500:next:char,b,k,f$   <ii>
1150 char,x+5,k-1,b$:char,15,22,a$
:a=10                               <dj>
1160 fort=1toa:p=p+1:sound2,800,5:
sound1,200,3:char,9,1,right$("   "+
wh$+str$(p),5)                      <fg>
1170 nextt                          <ld>
1180 return                         <mp>
1190 ifl=1then1300                  <kk>
1200 ifl=2then1310                  <ji>
1210 ifl=3then1320                  <ee>
1220 fort=1to3:sound1,700,10:sound
2,770,5:next:mm=mm-1:l=l+1:r=1:got
o580                                <jo>
1230 fort=750to100step-25:sound1,2
00+t,5:next                         <hb>
1240 char,3,22,"nochmal (j/n)"      <pg>
1250 getkeys$:ifs$="n"thenscnclr:e
nd                                  <mp>
1260 ifs$="j"thengoto100            <fi>
1270 l=3:mm=3:fort=1to3:sound1,700
,10:sound2,770,5:next:goto580       <ek>
1280 fort=1to5:sound1,739,15:sound
1,810,17:sound1,739,10:sound1,854,
10:next:scnclr                      <in>
1290 char,10,11,"g r a t u l a t i
 o n":end                           <nf>
1300 ifp=>80then1220:else1230       <pk>
1310 ifp=>160then1220:else1230      <ij>
1320 ifp=>240then1280:else1230      <dg>
1330 rem ======================     <hc>
1340 rem == titelmusik =========    <ca>
1350 rem ======================     <if>
1360 scnclr                         <jc>
1370 color0,6,1:char,17,2,a$        <om>
1380 char,11,1,wh$+"k a t e r   t o
 m "                                <jf>
1390 char,1,5,"sie muessen mit der
 katze 3 mauese-"                   <gj>
1400 char,1,6,"fallen setzen,die k
atze koennen sie "                  <eo>
1410 char,1,7,"mit den cursor tast
en rauf und runter"                 <fe>
1420 char,1,8,"steuern.haben sie 3
 fallen gesetzt,"                   <en>
1430 char,1,9,"so beginnt das spie
l.5 mauese kommen"                  <gf>
1440 char,1,10,"hintereinander aus
 ihren loechern."                   <nk>
1450 char,1,11,"laeuft eine maus i
n die falle,so er-"                 <hn>
1460 char,1,12,"halten sie 10 pukt
e.jeder level hat"                  <hj>
1470 char,1,13,"5 runden. um in le
vel 2 zu kommen"                    <pc>
1480 char,1,14,"muessen sie 80 pun
kte erreichen."                     <gp>
1490 char,1,15,"im level 2 kommen
dann nur noch "                     <pk>
1500 char,1,16,"4 maeuse,und sie b
rauchen 160 punkte"                 <lo>
1510 char,1,17,"um level3 zu errei
chen.dann kommen"                   <cj>
1520 char,1,18,"nur noch 3 maeuse
aus ihren loechern."                <ll>
1530 char,1,19,"haben sie in level
3 240 punkte er-"                   <ce>
1540 char,1,20,"reicht,so haben si
e gewonnen."                        <lk>
1550 char,1,21,"das setzen der fal
le erfolgt durch "                  <hl>
1560 char,1,22,"druecken der space
 taste!!"                           <gc>
1570 do:vol6                        <ko>
1580 reads,t                        <ff>
1590 sound1,s,t                     <le>
1600 loopuntils=0                   <ee>
1610 data739,15,810,17,739,10,810,
15,854,10,854,10,810,20,739,15,810
,10,854,10                          <pd>
1620 data834,15,739,15,739,15,739,
10,798,10,810,22                    <jm>
1630 data798,10,810,5,834,15,739,1
5,739,15,810,15,834,12,854,5        <oi>
1640 data864,15,854,15,834,15,810,
15,834,22,739,15,810,17,739,10,810
,15,854,15                          <jj>
1650 data810,20,739,15,810,12,854,
5,834,15,739,15,739,15,739,10,798,
10,810,20                           <cc>
1660 data798,12,810,5,834,20,739,1
5,881,15,854,20,810,15,854,15,834,
15,881,15                           <co>
1670 data834,15,881,15,810,15,810,
12,854,5,834,20,810,20,739,15,854,
15,810,20                           <lc>
1680 data0,0                        <mo>
1690 char,3,23,fl$+"zum spiel bitt
e eine taste druecken"+fo$          <nh>
1700 getkeyt$:return                <jk>
1710 rem nachspann =============    <ap>
1720 rem * farbcodes/steuercodes*   <nc>
1730 wh$=chr$(005):c4$=chr$(017)    <kb>
1740 re$=chr$(028):gr$=chr$(030)    <cm>
1750 oe$=chr$(129):fl$=chr$(130)    <bf>
1760 fo$=chr$(132):bk$=chr$(144)    <na>
1770 c1$=chr$(157)                  <ai>
1780 w$=c4$+c1$+c1$+c1$+c1$         <ki>
1790 return                         <jg>
1800 end                            <jo>
1810 rem ende                       <el>
```

PYRAMID GUY

Auf Schatzsuche

Im Innern einer Pyramide wartet ein Schatz, doch auch Gefahr. Der Sprung auf einen Kaktus oder das Zusammentreffen mit einem Monster kosten ein Leben.

Mit Joystick oder Tastendruck bewegen Sie sich durch die Gewölbe einer Pyramide. In jedem Raum müssen Sie sich einen Edelstein beschaffen, damit Sie den nächsten Raum betreten dürfen. Gefahr droht besonders von einem Monster, das die Gewölbewände durchdringt und Ihren Weg kreuzt. Da es Sie nicht verfolgt, brauchen Sie nur den Zusammenstoß vermeiden. Auch dürfen Sie nicht in einen Schacht springen, auf dessen Grund sich ein Kaktus befindet. Es stehen Ihnen zehn Leben zur Verfügung. Dies ist jedoch nicht besonders viel in Anbetracht von 32 Räumen. Nur nach längerer Übung werden Sie das Ziel erreichen.
Eine Besonderheit finden Sie im Programm ab Zeile 20000. Dort liegen die Bilddatas für die 32 Räume. Da in der Originalfassung reverse und nicht abdruckbare Zeichen vorlagen, mußten wir eine Verschlüsselung der Daten vornehmen. Eine Wandlung in Strings war nicht zulässig, da es sich um Datazeilen handelte. Die Decodierung findet bei der Ausgabe durch einen verbogenen BSOUT-Vektor statt.
Die Maschinenroutine können Sie betrachten, wenn Sie das Spiel unterbrechen, in den Maschinenmonitor gehen und mit **D139** auflisten. Ein Return wird von der Routine normal ausgegeben. Großbuchstaben von A bis P stellen den High-Nibble um, Kleinbuchstaben repräsentieren den Low-Nibble des auszugebenden Codes. Beispielsweise werden drei Leerzeichen mit **Caaa** ausgedrückt. Dabei wird hexadezimal 20 durch **Ca** repräsentiert. Für die folgenden Leerzeichen genügt ein kleines **a**, da der High-Nibble bereits durch das vorangegangene **C** richtig gesetzt wurde.
Da die Bilder jeweils aus acht Zeilen bestehen und schön hintereinander angeordnet sind, sollte es Ihnen nicht schwerfallen, eine eventuell fehlerhaft abgetippte Datazeile ausfindig zu machen. Im Menü des Programmes können Sie sich durch Auswählen des Punktes zwei, Demo Mode, alle Räume zeigen lassen und sich von deren Korrektheit überzeugen.

LADEANWEISUNG

Laden Sie den Loader ein und starten ihn. Die anderen Programmteile werden dann automatisch nachgeladen (gilt auch für Datasette).

SAVEANLEITUNG

Sie müssen, wenn das Programm auf Kassette abgespeichert werden soll, erst den Loader, danach den Code und zum Schluß das Hauptprogramm sichern (saven). Es brauchen, um es auf Kassette beziehungsweise Diskette zu sichern, keine Programmänderungen vorgenommen zu werden. Die Programmnamen dürfen nicht geändert werden, da die Programmteile sich nicht mehr selbständig laden würden. □

```
10 rem guy-loader===============p4 <pd>
20 rem (p) commodore welt team     <ho>
30 rem ========================== <ng>
40 rem (c) by mark luennamann      <lh>
50 rem      and michael sperling   <mh>
60 rem                             <ah>
70 rem basic v3.5                  <nl>
80 rem plus4 (c16/116 + 64 kb)     <fd>
90 rem ========================== <jg>
100 gosub 230                      <aj>
110 printcl$c4$c4$c4$wh$spc(11)"(c
) by m. & m.soft                   <cd>
120 printleft$(qd$,5)re$spc(14)"bi
tte warten                         <bk>
130 color0,1:color4,1:fora=1to1000
:next:color1,1                     <cc>
140 char,0,0,"monitor"             <gi>
150 char,0,6,"t d000 d800 1000"    <ml>
160 char,0,8,"x"                   <mg>
170 char,0,10,"poke43,0:poke44,24:
poke6143,0:new"                    <of>
180 char,0,13,"load"+chr$(34)+"guy
-code"+chr$(34)+","+str$(peek(174)
)                                  <mm>
190 char,0,18-(peek(174)=1),"run"  <ao>
200 poke1319,19                    <ol>
210 fori=1to6:poke1319+i,13:next   <po>
220 poke239,7:end                  <nj>
230 rem nachspann ===============  <nl>
240 rem * farbcodes/steuercodes *  <lo>
250 wh$=chr$(005):c4$=chr$(017)    <oo>
260 re$=chr$(028):cl$=chr$(147)    <co>
270 rem ********* zeichenfolgen *  <ie>
280 for q=1 to 40                  <oc>
290 qd$=qd$+c4$                    <en>
300 next q                         <ak>
310 return                         <pm>
320 rem ========================   <il>
330 rem  p r o g r a m m e n d e   <al>
340 rem ========================   <oc>
```

```
10 rem guy-code=================p4 <fi>
20 rem (p) commodore welt team     <ho>
30 rem ========================== <ng>
40 rem (c) by mark luennemann      <pl>
50 rem      and michael sperling   <mh>
60 rem                             <ah>
70 rem basic v3.5                  <nl>
80 rem c16/116/plus4               <ki>
90 rem ========================== <jg>
100 gosub 1140                     <dk>
110 printcl$c4$c4$c4$fl$spc(9)wh$"
ich "re$"poke "cy$"die "gr$"daten
"bl$"!!!"                          <ak>
120 data 60,66,165,129,126,36,36,1
02                                 <gc>
130 data 60,66,165,129,126,36,38,9
6                                  <ff>
```

```
140 data 60,66,141,129,126,36,36,5
4                                  <kj>
150 data 15,16,35,32,31,9,9,12     <me>
160 data 0,128,64,64,128,0,64,0    <hd>
170 data 0,1,2,2,1,0,0,0           <dl>
180 data 240,8,52,4,248,144,208,24 <ji>
190 data 3,4,8,8,15,2,3,0          <jf>
200 data 192,32,208,16,224,64,96,0 <eg>
210 data 60,66,177,129,126,36,36,1
08                                 <jj>
220 data 0,1,2,2,1,0,1,0           <km>
230 data 240,8,196,4,248,144,144,4
8                                  <fn>
240 data 15,16,44,32,31,9,11,24    <cf>
250 data 0,128,64,64,128,0,0,0     <op>
260 data 3,4,11,8,15,2,6,0         <on>
270 data 192,32,16,16,224,64,192,0 <ma>
280 data 24,36,90,129,189,153,129,
255                                <bc>
290 data 60,102,255,129,153,165,12
9,255                              <jo>
300 data 60,66,189,129,126,60,66,6
6                                  <bc>
310 data 0,0,60,66,165,129,126,36  <kk>
320 data 38,96,0,0,0,0,0,0         <og>
330 data 0,0,0,0,60,66,165,129     <je>
340 data 126,36,102,0,0,0,0,0      <kd>
350 data 0,0,0,0,0,0,60,66         <nc>
360 data 165,129,126,36,100,6,0,0  <ph>
370 data 0,24,60,102,102,60,24,0   <jo>
380 data 24,36,90,129,189,153,129,
255                                <je>
390 data 129,66,36,24,24,36,66,129 <kc>
400 data 129,66,60,36,36,60,66,129 <ll>
410 data 129,126,66,66,66,66,126,1
29                                 <fj>
420 data 129,126,66,90,90,66,126,1
29                                 <dl>
430 data 255,129,189,165,165,189,1
29,255                             <im>
440 data 0,126,66,90,90,66,126,0   <nh>
450 data 0,0,0,24,24,0,0,0         <lo>
460 data 0,0,60,36,36,60,0,0       <ai>
470 data 0,126,66,66,66,66,126,0   <pb>
480 data 255,129,129,129,129,129,1
29,255                             <hi>
490 data 66,90,126,90,24,24,24,0   <en>
500 data 0,0,0,3,12,48,192,192     <ki>
510 data 192,204,222,222,204,192,1
92,192                             <og>
520 data 48,12,3,1,1,1,1,1         <pd>
530 data 1,1,1,1,3,7,31,63         <kb>
540 data 15,48,192,0,124,146,124,1 <hm>
550 data 1,3,3,7,13,128,192,191    <lh>
560 data 64,32,31,192,176,143,128,
128                                <ma>
570 data 128,128,128,128,192,224,2
48,252                             <kl>
580 data 240,12,3,0,62,73,62,128   <ac>
590 data 128,192,192,224,176,1,3,2
53                                 <bo>
600 data 2,4,248,3,13,241,1,1      <oc>
610 data 0,0,0,192,48,12,3,3       <gb>
620 data 3,51,123,123,51,3,3,3     <bp>
630 data 12,48,192,128,128,128,128
,128                               <gf>
640 data 153,90,60,255,255,60,90,1
53                                 <oj>
650 data 90,165,90,189,189,90,165,
90                                 <hc>
660 data 219,36,219,36,219,36,219,
36                                 <ga>
670 data 165,90,165,90,90,165,90,1
65                                 <ln>
680 data 165,90,189,102,102,189,90
,165                               <mb>
690 data 129,126,102,90,90,102,126
,129                               <do>
700 data 126,219,165,219,219,165,2
19,126                             <nc>
710 data 170,85,170,85,170,85,170,
85                                 <ko>
720 data 36,36,231,24,24,231,36,36 <io>
730 data 189,102,231,129,129,231,1
02,189                             <eg>
740 data 126,153,153,255,255,153,1
53,126                             <lm>
750 data 36,36,102,255,255,102,36,
36                                 <oc>
760 data 153,126,66,66,66,66,126,1
53                                 <ad>
770 data 255,231,231,153,153,231,2
31,255                             <dp>
780 data 60,36,231,153,153,231,36,
60                                 <nd>
790 data 66,195,60,36,36,60,195,66 <cd>
800 data 126,189,219,231,231,219,1
89,126                             <bh>
810 data 195,231,126,60,60,126,231
,195                               <bl>
820 data 231,231,231,0,0,231,231,2
31                                 <ed>
830 data 126,255,231,195,195,231,2
55,126                             <na>
840 data 239,247,255,189,126,255,2
47,239                             <mp>
850 data 165,102,102,255,255,102,1
02,165                             <kk>
860 data 255,181,239,219,189,239,1
81,255                             <bp>
870 data 255,195,189,165,165,189,1
95,255                             <im>
880 data 255,165,219,255,255,219,1
65,255                             <ee>
890 data 36,60,255,126,126,255,60,
36                                 <im>
```

```
900 data 195,255,102,90,90,102,255
,195                               <bg>
910 data 231,152,165,90,90,165,152
,231                               <ha>
920 data 255,152,152,231,231,152,1
52,255                             <mp>
930 data 66,195,36,219,219,36,195,
66                                 <oo>
940 data 189,90,189,255,255,189,90
,189                               <ij>
950 data 90,255,102,219,219,102,25
5,90                               <go>
960 fori=1to63:forj=0to7           <oe>
970 readz:poke4608+8*i+j,z:nextj,i <kl>
980 fori=1to15:forj=0to7           <np>
990 readz:poke4352+8*i+j,z:nextj,i <ad>
1000 for i=1 to 6:for j=0to7       <kh>
1010 readz:poke4096+8*i+j,z:nextj,
i                                  <jn>
1020 poke 65298,192:poke 65299,16  <fp>
1030 printcl$left$(qd$,11)wh$spc(1
5)"your guy                        <op>
1040 printc4$re$spc(17)z6$za$ze$zh
$                                  <hl>
1050 printspc(17)z7$zb$zf$zi$      <lp>
1060 printspc(17)z8$zc$zg$zj$      <cd>
1070 printspc(17)z9$zd$z9$zd$      <pf>
1080 printbl$zz$                   <kc>
1090 printhe$bk$"load"chr$(34)"pyr
amid-guy"+chr$(34)","peek(174)     <if>
1100 char,0,5-(peek(174)=1),"run"  <ce>
1110 poke1319,19:poke1320,13       <ca>
1120 poke1319,19:poke1320,13       <hd>
1130 poke1321,13:poke239,3:end     <dk>
1140 rem nachspann =============== <hb>
1150 rem * farbcodes/steuercodes * <gi>
1160 wh$=chr$(005):c4$=chr$(017)   <lg>
1170 he$=chr$(019):re$=chr$(028)   <ni>
1180 gr$=chr$(030):bl$=chr$(031)   <ag>
1190 fl$=chr$(130):bk$=chr$(144)   <ff>
1200 cl$=chr$(147):cy$=chr$(159)   <pd>
1210 rem *** zeichensatz/graphik * <mj>
1220 z2$=chr$(163):z6$=chr$(167)   <ph>
1230 z7$=chr$(168):z8$=chr$(169)   <ll>
1240 z9$=chr$(170):za$=chr$(171)   <ok>
1250 zb$=chr$(172):zc$=chr$(173)   <mb>
1260 zd$=chr$(174):ze$=chr$(175)   <cj>
1270 zf$=chr$(176):zg$=chr$(177)   <ba>
1280 zh$=chr$(178):zi$=chr$(179)   <om>
1290 zj$=chr$(180)                 <ei>
1300 rem ********* zeichenfolgen * <kk>
1310 for q=1 to 40                 <od>
1320 qd$=qd$+c4$:zz$=zz$+z2$       <mp>
1330 next q                        <dc>
1340 return                        <ba>
1350 rem ========================= <ei>
1360 rem  p r o g r a m m e n d e  <el>
1370 rem ========================= <pk>
```

```
10 rem pyramid-guy==============p4 <nl>
20 rem (p) commodore welt team     <ho>
30 rem =========================== <ng>
40 rem (c) by mark luennemann      <pl>
50 rem     and michael sperling    <mh>
60 rem                             <ah>
70 rem basic v3.5                  <nl>
80 rem plus4 (c16/116  + 64kb)     <fd>
90 rem =========================== <jg>
100 gosub 4400                     <fg>
110 :                              <bg>
120 rem **** goto menu ****        <ag>
130 :                              <dp>
140 gosub4310:poke 65298,192:poke
65299,16                           <lj>
150 vol 8:for n=600 to 1000 step 1
0                                  <nm>
160 sound 1,n,3                    <ji>
170 next n                         <pn>
180 for n=1 to 8:key n,"":next n   <co>
190 ss=1:cl=1:goto 2980            <ao>
200 poke 65286,peek(65286) and 239 <jf>
210 color 1,7,4:scnclr:char ,10,8,
"))))))))))))))))))))"             <fn>
220 for n=8 to 16                  <fo>
230 char ,9,n,")":char ,30,n,")"   <bj>
240 next n:char ,9,17,")))))))))))
)))))))))))"                       <pj>
250 :                              <cp>
260 goto 2680                      <ec>
270 goto 2070                      <gl>
280 :                              <gm>
290 rem * joystick and keyboard *  <ei>
300 :                              <je>
310 goto 1750                      <hg>
320 if cl=0 then 400               <hb>
330 if joy(1)=7   then 590         <id>
340 if joy(1)=3   then 740         <jb>
350 if joy(1)=134 then 890         <jk>
360 if joy(1)=136 then 1080        <od>
370 if joy(1)=130 then 1510        <ll>
380 if joy(1)=132 then 1320        <kh>
390 goto 470                       <dc>
400 get a$                         <lj>
410 if a$="o" then 590             <ml>
420 if a$="p" then 740             <lj>
430 if a$="z" then 890             <nh>
440 if a$="a" then 1080            <gm>
450 if a$="s" then 1510            <be>
460 if a$="x" then 1320            <om>
470 get u$:if u$=" " then 2570     <ma>
480 if u$="n" then 2980            <ck>
490 if u$="m" then 2250            <cf>
500 if hh=0 and d=1 then sound 1,9
90,5:hh=1                          <nf>
510 :                              <dj>
520 rem **** foot movement ****    <ej>
530 :                              <gc>
```

```
540 if s=1 then poke(r-1024),74:po
ke r,65:s=0:sound 1,500,1:goto 175
0                                 <oh>
550 poke(r-1024),74:poke r,66:s=1:
goto 310                          <lb>
560 :                             <jo>
570 rem **** go left ****         <om>
580 :                             <mg>
590 r=r-1                         <io>
600 if peek(r)=32  then 640       <na>
610 if peek(r)=90  then d=1:goto 6
40                                <ab>
620 if peek(r)=91  then 2050      <ih>
630 if peek(r)<>32 then r=r+1:goto
 310                              <nl>
640 poke(r+1),74:poke (r-1024),74 <kl>
650 poke r,75:poke(r+1),76        <en>
660 poke r,79:poke(r+1),80        <cp>
670 sound 1,700,3                 <ja>
680 poke r,77:poke(r+1),78        <fk>
690 poke(r+1),32:poke r,74        <hd>
700 goto 310                      <ip>
710 :                             <ml>
720 rem **** go right ****        <ji>
730 :                             <pd>
740 r=r+1                         <fb>
750 if peek(r)=32  then 790       <eh>
760 if peek(r)=90  then d=1:goto 7
90                                <fm>
770 if peek(r)=91  then 2050      <dm>
780 if peek(r)<>32 then r=r-1:goto
 310                              <ol>
790 poke(r-1),67:poke (r-1024),74 <bh>
800 poke(r-1),68:poke r,69        <oc>
810 poke(r-1),72:poke r,73        <jf>
820 sound 1,740,3                 <ff>
830 poke(r-1),70:poke r,71        <dj>
840 poke(r-1),32:poke r,67        <mk>
850 goto 310                      <pi>
860 :                             <pi>
870 rem **** jump left:small ***** <ao>
880 :                             <ca>
890 if peek(r-41)=90  then d=1:got
o 920                             <ec>
900 if peek(r-41)=91  then 2050   <ge>
910 if peek(r-41)<>32 then 1750   <hj>
920 r=r-41:poke(r+41),32:poke (r-1
024),74:poke r,74                 <fb>
930 sound 1,740,3                 <jn>
940 if peek(r-1)=90   then d=1:got
o 970                             <dn>
950 if peek(r-1)=91   then 2050   <pj>
960 if peek(r-1)<>32  then 1750   <ma>
970 r=r-1:poke(r+1),32:poke (r-102
4),74:poke r,74                   <dp>
980 sound 1,800,3                 <od>
990 if peek(r+39)=90  then d=1:got
o 1020                            <ig>
1000 if peek(r+39)=91  then 2050  <bl>
1010 if peek(r+39)<>32 then 1750  <kn>
1020 r=r+39:poke(r-39),32:poke (r-
1024),74:poke r,74                <pn>
1030 sound 1,900,3                <gk>
1040 goto 1750                    <bl>
1050 :                            <hg>
1060 rem **** jump left :big **** <je>
1070 :                            <jo>
1080 if peek(r-40)=90  then d=1:go
to 1110                           <im>
1090 if peek(r-40)=91  then 2050  <je>
1100 if peek(r-40)<>32 then 1750  <pa>
1110 r=r-40:poke(r+40),32:poke (r-
1024),74:poke r,74                <hh>
1120 sound 1,740,3                <fm>
1130 if peek(r-40)=90  then d=1:go
to 1160                           <ee>
1140 if peek(r-40)=91  then 2050  <cl>
1150 if peek(r-40)<>32 then 1750  <pk>
1160 r=r-40:poke(r+40),32:poke(r-1
024),74:poke r,74                 <ll>
1170 sound 1,800,3                <kg>
1180 if peek(r-41)=90  then d=1:go
to 1210                           <oc>
1190 if peek(r-41)=91  then 2050  <gb>
1200 if peek(r-41)<>32 then 1750  <dd>
1210 r=r-41:poke(r+41),32:poke(r-1
024),74:poke r,74                 <gf>
1220 sound 1,900,3                <ki>
1230 if peek(r+39)=90  then d=1:go
to 1260                           <ie>
1240 if peek(r+39)=91  then 2050  <bc>
1250 if peek(r+39)<>32 then 1750  <pj>
1260 r=r+39:poke(r-39),32:poke(r-1
024),74:poke r,74                 <hf>
1270 sound 1,1000,3               <gk>
1280 goto 1750                    <di>
1290 :                            <fi>
1300 rem **** jump right:small ***
*                                 <on>
1310 :                            <ia>
1320 if peek(r-39)=90  then d=1:go
to 1350                           <af>
1330 if peek(r-39)=91  then 2050  <oh>
1340 if peek(r-39)<>32 then 1750  <he>
1350 r=r-39:poke(r+39),32:poke(r-1
024),74:poke r,67                 <ep>
1360 sound 1,740,3                <gl>
1370 if peek(r+1)=90   then d=1:go
to 1400                           <en>
1380 if peek(r+1)=91   then 2050  <gp>
1390 if peek(r+1)<>32  then 1750  <ba>
1400 r=r+1:poke(r-1),32:poke(r-102
4),74:poke r,67                   <cg>
1410 sound 1,800,3                <nd>
1420 if peek(r+41)=90  then d=1:go
to 1450                           <bb>
```

```
1430 if peek(r+41)=91  then 2050      <ji>
1440 if peek(r+41)<>32 then 1750      <kb>
1450 r=r+41:poke(r-41),32:poke(r-1
024),74:poke r,67                     <dk>
1460 sound 1,900,3                    <nj>
1470 goto 1750                        <ab>
1480 :                                <nf>
1490 rem **** jump right:big ****     <el>
1500 :                                <pn>
1510 if peek(r-40)=90  then d=1:go
to 1540                               <cb>
1520 if peek(r-40)=91  then 2050      <li>
1530 if peek(r-40)<>32 then 1750      <bi>
1540 r=r-40:poke(r+40),32:poke(r-1
024),74:poke r,74                     <fj>
1550 sound 1,740,3                    <cp>
1560 if peek(r-40)=90  then d=1:go
to 1590                               <ha>
1570 if peek(r-40)=91  then 2050      <pl>
1580 if peek(r-40)<>32 then 1750      <de>
1590 r=r-40:poke(r+40),32:poke(r-1
024),74:poke r,74                     <mk>
1600 sound 1,800,3                    <dg>
1610 if peek(r-39)=90  then d=1:go
to 1640                               <nn>
1620 if peek(r-39)=91  then 2050      <ka>
1630 if peek(r-39)<>32 then 1750      <nn>
1640 r=r-39:poke(r+39),32:poke(r-1
024),74:poke r,74                     <cd>
1650 if peek(r+41)=90  then d=1:go
to 1690                               <ha>
1660 sound 1,900,3                    <ac>
1670 if peek(r+41)=91  then 2050      <kd>
1680 if peek(r+41)<>32 then 1750      <ll>
1690 r=r+41:poke(r-41),32:poke(r-1
024),74:poke r,74                     <dn>
1700 sound 1,1000,3                   <mj>
1710 goto 1750                        <ch>
1720 :                                <lh>
1730 rem **** fall down ****          <ij>
1740 :                                <np>
1750 ee=0                             <jm>
1760 if peek(r+40)=32  then 1810      <jn>
1770 if peek(r+40)=90  then d=1:go
to 1810                               <ed>
1780 if peek(r+40)=91  then 2050      <hh>
1790 if peek(r+40)=102 then 2250      <hk>
1800 if peek(r+40)<>32 then o=1:go
to 1920                               <pf>
1810 r=r+40                           <in>
1820 ee=ee+38                         <aa>
1830 sound 1,ee+700,3                 <eh>
1840 poke(r-40),65:poke(r-1024),74 <fh>
1850 poke(r-40),84:poke r,85          <nm>
1860 poke(r-40),86:poke r,87          <ca>
1870 poke(r-40),88:poke r,89          <fa>
1880 poke(r-40),32:poke r,65          <eo>
1890 :                                <am>
1900 rem **** monster movement ***
*                                     <ma>
1910 :                                <de>
1920 if e=0 then 1950                 <bo>
1930 e=0:p=3400:t=41                  <el>
1940 i=int(rnd(1)*19)+2               <fn>
1950 p=p+40                           <fk>
1960 l=p+i                            <cf>
1970 poke(l-40),t:t=peek(l)           <nf>
1980 poke l,83                        <ji>
1990 if t<90 and t>64 then 2250       <np>
2000 if p=>3700 then e=1:poke l,t     <pi>
2010 if o=1 then o=0:goto 320         <ja>
2020 goto 1770                        <ag>
2030 :                                <cf>
2040 rem **** new picture delivers
 ****                                 <kf>
2050 :                                <eo>
2060 if d=0 then 2250:else d=0:e=1 <bb>
2070 poke 65286,peek(65286) and 23
9                                     <ed>
2080 f=f+100:if f>23120 then 2810     <io>
2090 restore f                        <ch>
2100 gosub4380:for n=1 to 8           <pe>
2110 read bb$(n)                      <ki>
2120 char,10,n+8,"":printbb$(n)       <oc>
2130 next n:gosub4390                 <nh>
2140 read r:hh=0                      <ke>
2150 for w=500 to 900 step 100        <ep>
2160 sound 1,w,5                      <li>
2170 next w                           <gg>
2180 g=g+1:char ,10,23,"room :"+st
r$(g)                                 <le>
2190 char ,10,20,"guys :"+str$(m)     <nj>
2200 poke 65286,peek(65286) or 16     <jj>
2210 goto 310                         <fe>
2220 :                                <kd>
2230 rem **** loose life ****         <el>
2240 :                                <ml>
2250 m=m-1                            <hi>
2260 sound 1,600,10:sound 2,800,10 <nh>
2270 sound 1,900,10:sound 2,600,10 <ob>
2280 sound 1,450,10:sound 2,600,10 <hg>
2290 for k=92 to 101                  <lj>
2300 poke r,k                         <oj>
2310 for j=1 to 20:next j:next k      <oi>
2320 for k=101 to 98 step-1           <ji>
2330 for j=1 to20:next j              <eh>
2340 poke r,k                         <hi>
2350 next k                           <cm>
2360 sound 1,900,10:sound 2,750,10 <pd>
2370 sound 1,1000,10:sound 2,650,1
0                                     <ee>
2380 if m=0 then 2730                 <ln>
2390 char ,10,20,"guys :"+str$(m)     <gb>
2400 :                                <am>
2410 rem * old picture delivers *     <mh>
2420 :                                <de>
```

```
2430 poke 65286,peek(65286) and 23
9                                  <jh>
2440 restore f:e=1:d=0             <nb>
2450 gosub4380:for n=1 to 8        <pk>
2460 read bb$(n)                   <ji>
2470 char,10,n+8,"":printbb$(n)    <nf>
2480 next n:gosub4390              <hh>
2490 read r:hh=0                   <fe>
2500 char ,10,23,"room :"+str$(g)  <ib>
2510 char ,10,20,"guys :"+str$(m)  <he>
2520 poke 65286,peek(65286) or 16  <jm>
2530 goto 310                      <ga>
2540 :                             <cf>
2550 rem **** wait time ****       <kf>
2560 :                             <eo>
2570 for y=1 to 40                 <gb>
2580 next y                        <ng>
2590 poke r,65                     <fj>
2600 for y=1 to 40                 <fk>
2610 next y                        <eo>
2620 poke r,66                     <bh>
2630 get u$:if u$=chr$(13) then 31
0                                  <cn>
2640 goto 2570                     <ni>
2650 :                             <ac>
2660 rem **** graphic on ****      <ha>
2670 :                             <ck>
2680 poke 65298,192:poke 65299,16  <ki>
2690 goto 270                      <hk>
2700 :                             <gh>
2710 rem **** guy is died  ****    <kh>
2720 :                             <ip>
2730 for n=1 to200:next n:scnclr:p
rint " schade! aber du musst noch
viel ueben "                       <bg>
2740 poke 65298,196:poke 65299,209 <ki>
2750 print:print:print " bitte tas
te druecken"                       <hk>
2760 poke 239,0:getkey w$          <jn>
2770 goto 2980                     <dl>
2780 :                             <ah>
2790 rem ** guy found treasure **  <ap>
2800 :                             <cp>
2810 for n=1 to 200:next:scnclr    <cf>
2820 poke 65286,peek(65286) or 16  <od>
2830 poke 65298,196:poke 65299,209 <ck>
2840 print:print                   <im>
2850 print " sie haben guy zu dem
schatzgefuehrt":print              <oi>
2860 print " es ist nicht zu glaub
en aber sie haben":print           <pd>
2870 print " den fluch des pharao
gebrochen.und":print               <cm>
2880 print " das monster war ihnen
 unterlegen.das":print             <oe>
2890 print " monster wuenscht eine
 revange von":print                <ke>
2900 print " ihnen.":print:print   <me>
```

```
2910 print:print " bitte taste dru
ecken"                             <ik>
2920 poke 239,0:getkey w$          <jg>
2930 goto 2980                     <ep>
2940 :                             <ej>
2950 if f=23120 then 2980          <kj>
2960 rem **** menu variablen ****  <gl>
2970 :                             <if>
2980 scnclr:aa=0:f=19920:m=9:d=0   <in>
2990 poke 65298,196:poke 65299,208 <pf>
3000 e=1::g=0:hh=0                 <fg>
3010 :                             <nf>
3020 rem **** program menu ****    <pa>
3030 :                             <pn>
3040 color 0,1:color 4,1:color1,2  <jb>
3050 scnclr                        <nd>
3060 char ,14,1,bl$+"pyramid guy"  <cn>
3070 char ,9,7,ye$+"1 instructions
"                                  <hi>
3080 char,9,9,"2 demo mode"+b3$+"(
q=break)"                          <lb>
3090 char,9,11,"3 control"+b5$+"(j
oystick)"                          <ab>
3100 char,9,13,"4 sound"+b4$+b3$+"
(on)"                              <ln>
3110 char ,9,15,"7 goto system (65
529)"                              <fk>
3120 char ,9,17,"0 start game"     <ee>
3130 if cl=0 then char ,23,11,"(ke
yboard)"                           <ma>
3140 if cl=1 then char ,23,11,"(jo
ystick)"                           <mh>
3150 sound 1,890,20:sound 2,1010,1
0                                  <cg>
3160 if ss=1 then char ,23,13,"(on
) "                                <mm>
3170 if ss=0 then char ,23,13,"(of
f)"                                <pp>
3180 :                             <ck>
3190 :                             <dp>
3200 rem **** menu ****            <on>
3210 :                             <gh>
3220 get w$                        <fm>
3230 if w$="1" then 3340           <gb>
3240 if w$="2" then 3880           <nk>
3250 if w$="3" then 4140           <cc>
3260 if w$="4" then 4230           <af>
3270 if w$="7" then sys65529       <na>
3280 if w$="0" then sound 1,444,5:
goto 200                           <ga>
3290 if joy(1)=128 then 200        <ne>
3300 goto 3180                     <jn>
3310 :                             <cp>
3320 rem **** instructions ****    <mc>
3330 :                             <fi>
3340 scnclr:sound 3,990,10         <lg>
3350 print                         <np>
3360 print re$" guy ist ein wissen
```

```
schaftler.":print                         <pk>
3370 print " er hat die aufgabe in
 einer pyramide":print                    <kc>
3380 print " den schatz,dem ein ph
arao gehoerte,zu":print                   <oi>
3390 print " suchen.dabei muss er
allerlei gefahren":print                  <an>
3400 print " ausweichen.der konstr
ukteur der pyra-":print                   <ck>
3410 print " mide hat viele geheim
gaenge,in die":print                      <cf>
3420 print " raeume der pyramide,e
inbauen lassen.":print                    <jh>
3430 print " eine gefahr ist das m
onster das die":print                     <pk>
3440 print " kraft besitzt durch w
aende und tueren":print                   <le>
3450 print " zu gehen.dieses monst
er verfolgt den":print                    <fp>
3460 print " guy um ihn zu toeten!
 viel spass!!":print                      <mp>
3470 print " bitte taste druecken"        <po>
3480 getkey w$:sound 3,990,10             <nc>
3490 scnclr                               <eh>
3500 print                                <am>
3510 print cy$" guy besitzt sechs
arten der fortbe-":print                  <mp>
3520 print " wegung.":print               <el>
3530 print " 1 rechts gehen"              <if>
3540 print " 2 links gehen"               <mj>
3550 print " 3 kleiner sprung nach
 rechts"                                  <bc>
3560 print " 4 grosser sprung nach
 rechts"                                  <ij>
3570 print " 5 kleiner sprung nach
 links "                                  <bo>
3580 print " 6 grosser sprung nach
 links ":print                            <cc>
3590 print " diese werden durch jo
ystick oder key-":print                   <ag>
3600 print " board betaetigt":prin
t                                         <nc>
3610 print " 1 rechts"b$b3$b3$"ode
r taste p"                                <ia>
3620 print " 2 links"b$b4$b3$"oder
 taste o"                                 <cd>
3630 print " 3 fire und rechts-unt
en oder taste x"                          <cj>
3640 print " 4 fire und rechts-obe
n"b2$"oder taste s"                       <no>
3650 print " 5 fire und links-unte
n"b2$"oder taste z"                       <fi>
3660 print " 6 fire und links-oben
"b3$"oder taste a"                        <fk>
3670 print:print" bitte taste drue
cken"                                     <pc>
3680 getkey w$:sound 3,990,10             <oe>
3690 scnclr                               <ni>
3700 print                                <jo>
3710 print pu$" die taste space fu
ert in den pause-":print                  <fd>
3720 print " modus.durch druecken
von return wird":print                    <cl>
3730 print " das spiel wieder vort
gefuehrt.":print                          <dh>
3740 print " beim laengeren drueck
en auf die taste":print                   <lc>
3750 print " n gelangt man wieder
in das menu ":print                       <ga>
3760 print " mit der taste m werde
n guy die himmels-":print                 <hg>
3770 print " tueren geoeffnet.":pr
int                                       <hi>
3780 print " ohne einen ringdiaman
ten kommen sie":print                     <io>
3790 print " nicht lebendig aus de
m raum.":print                            <ie>
3800 print " m&m soft wuenscht ihn
en viel spass mit":print                  <jf>
3810 print " diesem abenteuerspiel
.":print                                  <hj>
3820 print " bitte taste druecken"        <le>
3830 getkey w$:sound 3,990,10             <fm>
3840 goto 2980                            <jb>
3850 :                                    <gm>
3860 rem **** demo mode ****              <lj>
3870 :                                    <je>
3880 poke 65286,peek(65286) and 23
9                                         <ne>
3890 g=0:scnclr:char ,10,8,bl$+rn$
+b$+b$+rf$                                <ml>
3900 for n=8 to 16                        <jh>
3910 char ,9,n,rn$+" "+rf$:char ,3
0,n,rn$+" "+rf$                           <ph>
3920 next n:char ,9,17,rn$+b$+b$+b
2$+rf$:f=19920                            <kh>
3930 poke 65298,192:poke 65299,16         <mc>
3940 poke 65286,peek(65286) and 23
9                                         <cp>
3950 f=f+100:restore f                    <na>
3960 gosub4380:for n=1 to 8               <ln>
3970 read bb$(n)                          <jg>
3980 char,10,n+8,"":printbb$(n)           <cm>
3990 next n:gosub4390                     <il>
4000 read r                               <bp>
4010 for w=500 to 1000 step 100           <nj>
4020 sound 1,w,5                          <ib>
4030 next w                               <jb>
4040 g=g+1:char ,10,23,"room :"+st
r$(g)                                     <id>
4050 poke 65286,peek(65286) or 16         <jm>
4060 for w=1 to 100                       <jn>
4070 get w$:if w$="q" then 2980           <pj>
4080 next w                               <fk>
4090 if f=23120 then 2980                 <ji>
4100 goto 3940                            <da>
```

```
4110 :                              <hg>
4120 rem **** control ****          <fh>
4130 :                              <jo>
4140 if cl=1 then cl=0:goto 4180    <kn>
4150 cl=1                           <ee>
4160 char ,23,11,"(joystick)"       <hm>
4170 sound 1,600,10:goto 3180       <po>
4180 char ,23,11,"(keyboard)"       <bc>
4190 sound 1,900,10:goto 3180       <oe>
4200 :                              <ck>
4210 rem **** sound ****            <aj>
4220 :                              <fd>
4230 if ss=1 then ss=0:else goto 4
270                                 <de>
4240 sound 1,900,5:vol 0            <ge>
4250 char ,23,13,"(off)"            <ci>
4260 goto 3180                      <cf>
4270 ss=1                           <gi>
4280 sound 1,900,5:vol 8            <cn>
4290 char ,23,13,"(on) "            <mi>
4300 goto 3180                      <cc>
4310 restore4330:fori=313to341      <gk>
4320 reada:pokei,a:next:restore:re
turn                                <hp>
4330 data 201,013,240,022,201,193   <kk>
4340 data 048,012,056,233,193,010   <pb>
4350 data 010,010,010,141,056,001   <fe>
4360 data 024,096,056,233,065,013   <pi>
4370 data 056,001,076,075,236       <na>
4380 poke312,255:poke804,57:poke80
5,1:return                          <lc>
4390 poke804,75:poke805,236:return <mh>
4400 rem nachspann ============== <jd>
4410 rem * farbcodes/steuercodes * <ed>
4420 rn$=chr$(018):re$=chr$(028)    <ao>
4430 bl$=chr$(031):rf$=chr$(146)    <lf>
4440 pu$=chr$(156):ye$=chr$(158)    <oi>
4450 cy$=chr$(159):b$=chr$(32)      <jm>
4460 b2$=b$+b$:b3$=b2$+b$           <eg>
4470 b4$=b3$+b$:b5$=b4$+b$          <kf>
4480 b$=b5$+b5$:return              <hg>
4490 rem ======================== <gk>
20000 rem **** the picture ****     <bl>
20010 :                             <ck>
20020 data"BmKeCaBpNkCaaaaaaaaaaaa
BmKeCaaKee                          <eh>
20030 data"CaKeeCaaaKeeeeeeeeCaKeC
aaaBmNlBm                           <co>
20040 data"CaaaaaKeeCaaaaaaaaKeCaa
Kee                                 <op>
20050 data"eeeCaaKeeCaKeeeeeeeeCaa
aa                                  <km>
20060 data"KeeeeCaKeeCaaaaaaaaaKeC
aaa                                 <kc>
20070 data"KeeeCaaaKeeeeeCaKeeeCaa
aaa                                 <ol>
20080 data"KeeeCaaaaKeeeeCaKeeeeCa
Keee                                <dk>
20090 data"eeCaaJfKgBmCaaaKeeeCaaa
aaaKeee                             <pk>
20100 data 3482                     <ip>
20110 :                             <pd>
20120 data"BpNpppppCaaaaaNpppppppp
pp                                  <pa>
20130 data"CaaaNpCaaNpppCaNppCaaNp
ppCaNpp                             <cb>
20140 data"ppCaNppCaaaNpCaNpCaaaaB
pNkBpCaaNpp                         <ll>
20150 data"CaaaNppppCaNpCaNpCaNpCa
NpCaaaaNp                           <mc>
20160 data"CaNppCaaaaNpCaaaNpCaaaN
pCaaa                               <ik>
20170 data"aaaNpCaNpCaaNpppppCaNpC
aNppCaa                             <oo>
20180 data"NppCaNpCaNppppBmNlBpCaN
ppppppppCa                          <jj>
20190 data"NpCaaaaNpppppCaaaaaaaaa
a                                   <fm>
20200 data 3482                     <ad>
20210 :                             <lk>
20220 data"BoKfffffffBpNkBoKffffff
fffffBmNlBo                         <jp>
20230 data"KffffffCaaaKfffffffffCa
a                                   <fb>
20240 data"KfffCaKffCaaaKfffffCaKf
ffCaa                               <jd>
20250 data"KffCaaaaaaaaaaaaaaaaaa  <bg>
20260 data"KfCaaaKfffCaKfCaKfffCaK
fffffCa                             <fa>
20270 data"aaaaaKfCaaKfCaaaaaKffff
fCa                                 <fj>
20280 data"KfffCaaKfCaaKffffffffff
fCa                                 <hm>
20290 data"aaaJfKgBoCaKfCaJfKgBoCa
KffffffffffJfKgBo                   <ik>
20300 data 3722                     <li>
20310 :                             <if>
20320 data"AfNnnnnnnnnnnnnnnnnnnnn <kn>
20330 data"nnnnnnnCaaNnnnnnnnnCaNn
n                                   <op>
20340 data"BmNlAfCaaaaNnnnnCaaaaaa
aaNnCaNn                            <ga>
20350 data"nCaaaaaaaNnnnnnnnnnnCaa <eg>
20360 data"NnJfKgAfNnCaNnnnnCaaaaa
aaaaaaa                             <pg>
20370 data"NnnnCaaaBpNkAfNnnnnnnnn
nnnnCa                              <hd>
20380 data"NnnCaaNnnnnCaaaaaaaaaaa
a                                   <hi>
20390 data"NnnnnnnnnnnnnnnnnnnCaNn <eb>
20400 data 3691                     <do>
20410 :                             <em>
20420 data"JoPppppppppCaPppppppppp
pp                                  <ck>
20430 data"CaaaPppCaPpCaaaPppppCaPp
CaPpCaaBmNlJo                       <li>
```

```
20440 data"CaaaaaaPpCaPpCaPpCaPpCa
PpCaaaPpp                       <kp>
20450 data"pCaPpppCaaaPpCaPpCaPpCa
PpCaPpCaaa                      <lb>
20460 data"PpCaPpCaaaaPppCaaaPpCaa
aPpCaPpCa                       <ca>
20470 data"PpCaPppCaPpCaaPpCaPpCaa
aPpppppCa                       <mi>
20480 data"PpCaPpBpNkJoCaaaaaaPppC
aaPpCaaaaa                      <df>
20490 data"PpJfKgJoPppppJfKgJoPppC
aPpppCaaaPppCaPp                <bb>
20500 data 3482                 <mb>
20510 :                         <bg>
20520 data"JjCaaaaaaaaaaaaaaaaaaaK
b                               <fb>
20530 data"CaKbCaKbCaaKbCaaKbCaaKb
CaaKbbbCaa                      <ma>
20540 data"KbbCaKbAfCaKhlpLcJjKbbC
aKbCaKbbbbbCa                   <op>
20550 data"KbbCaKbbAfKimLadJjKbCaa
KbCaaKbbbCaa                    <go>
20560 data"KbbbbbAfKjnLbeJjKbCaKbb
bCaKbbbCaKb                     <mp>
20570 data"bbbbbAfKkokoJjKbBpNkJjC
aKbCaaKbbbCaa                   <oa>
20580 data"KbbbbbbCaKbCaKbbJfKgJjK
bbCaKbbbbCa                     <bg>
20590 data"BmNlCaaaaaaaaaaaaaaaaaa
a                               <if>
20600 data 3482                 <em>
20610 :                         <np>
20620 data"JmLmmmmCaaaaaaaaaaaaaaa
a                               <bi>
20630 data"LmmCaaaLmCaaLmCaLmCaaLm
CaaLmCaa                        <pi>
20640 data"LmCaaLmmmmCaLmmmCaLmCaa
aaLmmCa                         <mi>
20650 data"aaLmCaaaaaLmmmmmCaaLmCa
aLmCa                           <hh>
20660 data"aLmmmmmmmmCaaaLmmCaLmmC
aLmCa                           <mc>
20670 data"LmCaaaaaLmCaLmCaLmCaaLm
CaaLmCaLmCa                     <mp>
20680 data"BmNlJmCaLmCaLmCaaaLmCaL
mmCaLmCaaaaaa                   <nf>
20690 data"aaaaaLmmCaaaLmmCaaaJfKg
BpNkJfKgCaa                     <oo>
20700 data 3602                 <lo>
20710 :                         <kg>
20720 data"JpLkkkkkCaaaaaLkCaLkkCa
aaaaa                           <lp>
20730 data"aaaaLkCaLkkkCaaaLkkCaLk
kCaLkCa                         <pk>
20740 data"LkCaLkCaLkCaaaLkCaLkCaa
aaaaaLkCa                       <fj>
20750 data"aaBpNkJpCaLkJfKgJpLkCaa
aLkCaLkkCaLkCaaLkCa             <jp>
20760 data"aaLkCaaaaaJfKgCaJpLkCaa
aaaaLkkCa                       <mb>
20770 data"LkCaLkkkkkkkkkkkkkkJfKg
JpLkkCa                         <pa>
20780 data"LkCaLkkkkkkkkkkkkkkkkkC
a                               <na>
20790 data"LkJfKgBmNlCaaaaaaaaaaaa
aaaaa                           <co>
20800 data 3484                 <hj>
20810 :                         <hb>
20820 data"BmNlBmCaaaLffffffffffff
fCaLff                          <bc>
20830 data"fffCaLfCaaaLfCaaLfCaaaL
ffCaLff                         <ac>
20840 data"CaaaLfCaaLfCaaaaaaLfCaa
aaaa                            <cg>
20850 data"LffffffffffCaLfffffffCa
Lf                              <gd>
20860 data"ffCaBpNkBmLfCaaaaLfCaaa
aaaaaaLf                        <pi>
20870 data"fCaaLffCaLfCaaLfCaLfCaL
fCaLffCaaLf                     <jm>
20880 data"fCaLfffCaLffCaLfCaLfCaL
fCaLffCaLfCa                    <ko>
20890 data"LfCaaaaaLffCaaaLfCaLfCa
LffCaLfCa                       <eh>
20900 data 3725                 <oo>
20910 :                         <dj>
20920 data"IbCaaaaLggCaaaaLggCaaaa
LgCaaLg                         <hd>
20930 data"CaLggCaaaaLggCaaaaLggCa
aaaa                            <kl>
20940 data"LgggggggggggCaLggggggCa
Lg                              <dh>
20950 data"CaaaaaaaaaaaaaaaBpNkIbC
aaaa                            <np>
20960 data"LggCaLggggggggggCaaLggg
gg                              <mo>
20970 data"Caaaaaaaaaaaaaaaaaaaa    <hd>
20980 data"LgggggggggggggCaaLgggCa
Lg                              <mf>
20990 data"BmNlCaaaaaaaaaaaaJfKgJf
KgCaaaaa                        <bc>
21000 data 3482                 <ho>
21010 :                         <ac>
21020 data"JfKgCaaaaaaaaaaaaaaaaaa
a                               <pm>
21030 data"BmNlJfLhCaaaLhhhhhhhhhh
hhhhCa                          <bg>
21040 data"LhhhhhBpNkJfCaaaLhCaaaa
aaaaaa                          <kn>
21050 data"LhCaaaLhhhhCaLhCaLhhhhC
aaaaa                           <og>
21060 data"aaLhCaaaaaLhhCaLhCaaaaL
hhhh                            <ip>
21070 data"CaLhhCaLhCaaLhhhCaLhCaL
hCaaaaLhh                       <mp>
21080 data"CaLhCaaLhhhCaaaJfKgJfLh
```

```
CaLhhhCaaaLh                              <eh>
21090 data"CaLhCaaaaaaaaaaaLhhhhCa
aLh                                       <cc>
21100 data 3722                           <am>
21110 :                                   <ml>
21120 data"JoLllllCaaaaaaaLlCaaaaa
aaa                                       <hd>
21130 data"aLlllCaLlllCaLlCaLlCaLl
BpNkJoLlCaaLlCa                           <gb>
21140 data"aLlllCaaaLlCaLlCaaaLllC
aLlCaLlJfKgJo                             <ia>
21150 data"CaaLllCaLlCaLlCaLlCaLlC
aaaaaaLlBmNlJo                            <ja>
21160 data"LlCaLllCaLlCaLlCaaaJfKg
JoLlCaaLlCaaaJfKgJo                       <dc>
21170 data"CaaaaaLlCaLlCaLlCaaaLlC
aaLlCaLll                                 <mf>
21180 data"llllllCaaaLlCaLlllCaLll
CaaLl                                     <cf>
21190 data"lllllllllCaLlCaaaaaaaaaL
l                                         <ol>
21200 data 3442                           <ib>
21210 :                                   <jc>
21220 data"BpLjjCaaaLjjCaaaLjCaLjj
CaaaLjjj                                  <lh>
21230 data"jjCaLjCaaLjCaLjCaaaaLjC
aLjCaaaLj                                 <hj>
21240 data"jCaaLjjCaLjCaLjjjjCaLjC
aLjCaLjCaLj                               <op>
21250 data"CaLjCaLjCaaaLjjCaaaaaaL
jCaaaJfKgBp                               <ka>
21260 data"LjjjCaaLjjCaaaLjCaaLjCa
aaaLjBmNlBp                               <ob>
21270 data"CaaaaaLjCaLjjCaLjjCaLjC
aLjjjjj                                   <pc>
21280 data"jjCaLjCaLjCaLjjJfKgBpLj
jCaLjCaLjjBpNkBpLjj                       <lo>
21290 data"jjCaLjCaaaLjjjjjjCaaaa
Ljj                                       <pc>
21300 data 3562                           <de>
21310 :                                   <fn>
21320 data"JgCaLiiiiiiCaaaaaaaaaa
aBmNlJg                                   <aj>
21330 data"CaLiiiiiiiiCaLiiiiiiiii
i                                         <mc>
21340 data"CaLiCaaaaaaLiiCaaaaaaaa
aa                                        <nf>
21350 data"aLiiiCaLiiCaaLiiCaaaLiC
aaaLiCa                                   <ii>
21360 data"aaaaaLiCaaaaLiiCaLiiiiC
aLiCa                                     <ai>
21370 data"LiCaLiCaaaaaaaaaLiCaaLi
iCaLiCa                                   <ha>
21380 data"LiCaLiCaLiCaLiiCaLiiCaL
iiCaLiiiiCa                               <ij>
21390 data"LiBpNkJgLiCaLiCaLiiJfKg
JgLiiCaaaLiiiiiJfKg                       <hg>
21400 data 3442                           <kg>
21410 :                                   <cf>
21420 data"BmNlJhCaaNnnnnnnCaNnCaN
nCaaaaNnCaa                               <co>
21430 data"NnnCaaaaaNnnCaNnCaaaNnn
CaNnCaNn                                  <oi>
21440 data"nCaNnnnnnnnCaNnCaaNnnnC
aaaa                                      <mk>
21450 data"NnCaaNnCaaaNnnCaNnCaaaa
NnCaaNnn                                  <fa>
21460 data"nCaaNnCaaaaaaNnnCaNnCaN
nnCaaa                                    <cb>
21470 data"NnnCaNnCaNnnCaNnnnnCaNn
CaaaNnnCa                                 <gc>
21480 data"NnnCaNnCaaNnCaNnCaaaaNn
CaNnCaaNnCa                               <jg>
21490 data"NnnCaaaaNnCaaaNnnnnnnnC
aBpNk                                     <he>
21500 data 3461                           <bf>
21510 :                                   <oo>
21520 data"BmNlJjCaaaaNpppppppCaaa
aNpppp                                    <bo>
21530 data"pCaNppCaNpCaNpppCaNpCaN
ppCaNpCaNpCa                              <lf>
21540 data"aNpppCaNpCaaaaaNpCaaaaN
pCaaa                                     <om>
21550 data"NpCaaNpCaaaNpCaNpCaNpCa
NppCaNpCaNpCa                             <db>
21560 data"aNpppppppCaNpCaNpCaaNpC
aNpCaNpCa                                 <co>
21570 data"aNpCaaaaaaaNpCaaaNppCaa
aNpCa                                     <fc>
21580 data"aNppppppppppppppppppCa     <fg>
21590 data"NpCaaaaaaBpNkCaaaaaaaaa
aaa                                       <ho>
21600 data 3644                           <nf>
21610 :                                   <lh>
21620 data"JlKaaaaaaaaaaaaaaaCaaaa
a                                         <bi>
21630 data"aaKaaaCaaaaaaKaaaaCaKaa
CaKa                                      <df>
21640 data"CaKaCaaaaKaaCaKaaCaKaaa
CaaKaCaa                                  <cl>
21650 data"aKaaaaCaaaaaaKaCaKaaCaa
aKaCa                                     <kd>
21660 data"aKaaBmNlJlKaaCaKaCaaaaa
KaaCaKaCaaa                               <gi>
21670 data"aKaaCaaKaCaKaaCaKaCaaaK
aCaaKaCaKa                                <bk>
21680 data"CaKaaaCaKaCaKaaCaaaKaCa
KaaCaKaCaa                                <da>
21690 data"aaaaaKaJfKgJlKaaJfKgJlC
aKaaCaaaaKaaBpNk                          <ji>
21700 data 3461                           <eb>
21710 :                                   <ho>
21720 data"JmCbaaaaaaaaaaaaaaaaaaB
pNk                                       <dg>
21730 data"JmCabbbbbbbbbbbbabaaaba <po>
21740 data"babbbbbbbbbbbabbbbba       <jj>
```

```
21750 data"abbaaaaaaaaabaaabbba       <nj>
21760 data"babababab bbaabbaaaba      <ee>
21770 data"bababbbbbbbaaabababa       <bj>
21780 data"aabaaaaaaaaabaaababa       <oh>
21790 data"abbBmNlJmCbabababababab
aaa                                   <in>
21800 data 3722                       <nm>
21810 :                               <ej>
21820 data"AfCaaaddddddadddaaaaa      <ck>
21830 data"adaadadaadaadBmNlAfCdad
dda                                   <of>
21840 data"aadaadaaaaaadaaaadad       <lp>
21850 data"aaadaddadddadddadaad       <jo>
21860 data"dadaaadaaaaadaaaddaa       <cc>
21870 data"dadadddadadadadadada       <le>
21880 data"aadadadadadadadddaaa       <ni>
21890 data"adBpNkAfCaaaaadJfKgAfCd
aaadddaaa                             <el>
21900 data 3722                       <hb>
21910 :                               <bb>
21920 data"JkCaEbbbbCaaaaaaEbbCaaE
bCaEbbb                               <ad>
21930 data"CaaaaaaEbbbCaaEbbCaaEbC
aEbCaEb                               <cp>
21940 data"CaEbbbbbCaaaaEbbbCaaaaa
aEb                                   <pp>
21950 data"CaEbbbbCaaEbCaaEbbbCaEb
bbCaaa                                <mm>
21960 data"aEbbbbJfKgJkCaEbCaaaaEb
CaEbCaaaaEb                           <mf>
21970 data"CaEbbbbCaEbbbbbCaEbCaEb
CaEbbCaEb                             <mf>
21980 data"CaEbbbbCaaEbbBpNkJkEbCa
EbCaEbCaaEbCaEb                       <ig>
21990 data"CaaaaaaaaaaEbCaaaaaaEbC
aBmNlJk                               <jd>
22000 data 3740                       <np>
22010 :                               <ni>
22020 data"JhCaaaaaEcccccccccCaEcc
ccBpNkJh                              <gb>
22030 data"CaEcCaaaaEcCaaaEcCaaaaa
EccCaa                                <ip>
22040 data"aEccCaEcCaEcCaEcCaEcCaE
cCaEccccCaEc                          <ni>
22050 data"CaEccCaEcCaaaEcCaEcCaEc
CaEccccCaEc                           <lm>
22060 data"CaEcCaaEccCaaaaaaaaaaaE
cCaa                                  <gf>
22070 data"aEcCaEccCaEcCaEccCaEccC
aEcCaaEccCa                           <mp>
22080 data"aEcCaEcCaaaaaaaaEcCaEcC
aaaEcCa                               <ll>
22090 data"BmNlCaJfKgJhEcCaEcccccc
CaEcCaEcCaEcCaaa                      <gl>
22100 data 3605                       <jd>
22110 :                               <kd>
22120 data"BmCaaaaaaaaaaaaaaEddddd
d                                     <ih>
22130 data"CaEdCaaEdCaEdBpNkBmEddd
CaaaEdddddd                           <lh>
22140 data"CaEdCaEddCaEddddddddCaaa
aEdCaEd                               <cc>
22150 data"CaEdCaaEdCaEdCaEdddCaaa
EddCaaaEd                             <cg>
22160 data"CaEddCaEdCaEdCaEdddCaEd
CaaaaEdCaa                            <lm>
22170 data"aEddCaaaEdCaEdddCaEdCaE
ddCaEddCa                             <de>
22180 data"aEddddddddddCaEdCaEdddC
aaa                                   <of>
22190 data"aaaaaaaaaBmNlCaJfKgBmEd
JfKgBmEdddddCa                        <em>
22200 data 3741                       <af>
22210 :                               <gk>
22220 data"JjEeCaaaaaaaaaaaaaaaaaa
a                                     <nl>
22230 data"EeCaEeeeeeeeeeeeeeeeeeC
a                                     <hh>
22240 data"aEeeeeeeCaaaaEeeCaaaaE
eCa                                   <gj>
22250 data"EeCaEeBmNlJjCaaaEeCaaEe
CaaEeCaaaEeeCa                        <dk>
22260 data"EeCaEeeeeCaaaEeCaaEeeCa
EeCaEeeCa                             <dk>
22270 data"aaEeeCaaaEeeeeCaaaaEeCa
aEeCa                                 <ob>
22280 data"aEeeeeeeeeeeCaEeeeeeCaE
eCa                                   <cf>
22290 data"aaaaaaaaaaEeCaaaaBpNkJj
EeCaaa                                <ab>
22300 data 3731                       <jm>
22310 :                               <de>
22320 data"JkEffffffffffCaaaaaaEfC
aEfCa                                 <dg>
22330 data"BmNlJkCaaEffCaaaEffCaEf
CaEffCaEfCaEfCa                       <kn>
22340 data"EfCaEffCaaaaaEfCaEfCaaa
aaaaa                                 <hn>
22350 data"EfCaEffCaEfffCaEfCaEfff
CaEfCaEfCaEf                          <mf>
22360 data"fCaaEfCaEfCaaaEfCaaaaaE
fCaEfCaEf                             <mk>
22370 data"ffCaEfCaEfCaEfCaaaEfffC
aEfCaaaa                              <mo>
22380 data"EffCaEfCaEfCaEfffCaaEfC
aaEffffCa                             <ca>
22390 data"EfCaaaaEfCaaaaaaaaaEfff
fBpNk                                 <dk>
22400 data 3579                       <di>
22410 :                               <pn>
22420 data"BoCaaaEggCaEgggggCaaaEg
BmNlBoCaEgCaEg                        <pp>
22430 data"CaEgBpNkBoCaEgCaaaaEggC
aEgCaEggCaaaa                         <ik>
22440 data"aEgCaEggCaEgCaaaEgCaaJf
KgBoEgggggCa                          <ih>
```

```
22450 data"aEggCaEgCaaaEgCaEggCaEg
gCaEgCaaa                         <lg>
22460 data"aEggCaEgggCaEgCaaaaEgCa
aaaaEg                            <gb>
22470 data"CaEggCaaaaaEggggggCaEgC
aEgCaEg                           <im>
22480 data"CaEggggggggCaaaaaaaEgCaE
gCaEg                             <il>
22490 data"CaaaaaaaaaEgggggggggg     <hi>
22500 data 3494                   <ki>
22510 :                           <me>
22520 data"JoCaaaaaaaLpppppppCaLpp
ppp                               <ji>
22530 data"CaLpppppCaaaLppCaaaLpCa
Lpppp                             <pc>
22540 data"CaBmNlJoCaaaLpppCaLppCa
aLppCaaLpCaLp                     <gi>
22550 data"CaLpCaaLpCaaLpCaLppCaLp
BpNkJoLpCaaaLpCa                  <ok>
22560 data"aLpCaLpCaaaLpCaaLpCaLpC
aLppCaLpCaa                       <af>
22570 data"aLpCaaaaaLppCaLpCaLpCaa
aJfKgCaJoCaa                      <oe>
22580 data"aLpCaaLppJfKggJoLpCaLpC
aLppppCaLpCaLp                    <dd>
22590 data"CaaaLpppJfKggJoLpCaaaaa
aaLpppCa                          <gh>
22600 data 3741                   <fm>
22610 :                           <ip>
22620 data"BmNlJfCakaaaaaakkaaaaa
aka                               <ea>
22630 data"aakakkkkkakkakakkaka   <ni>
22640 data"kaaaaakaakkaakkakaka   <ee>
22650 data"kakkakakaakakkkakJfKgJf
Cka                               <mj>
22660 data"aaakakaaaakaakaaaaka   <hg>
22670 data"kkakJfKgJfCaakkakaakakk
aaa                               <dk>
22680 data"BpNkJfCkakkkaakaakakaka
aaa                               <ck>
22690 data"aaaakkkakaJfKgCaaJfCkak
aaka                              <op>
22700 data 3501                   <li>
22710 :                           <fg>
22720 data"Caaaaaaaaaaaaaaaaaaa   <db>
22730 data"IbCallllllllallllllaalaa <ph>
22740 data"alllllllalaaaaalBpNkIbC
lal                               <fk>
22750 data"allllaaaalalllallllal   <bh>
22760 data"alllllllllaaaalaaaaaa   <oe>
22770 data"aaaaBmNlCaIbCaaaalaaala
lala                              <ao>
22780 data"allllallaalalaaalala   <ch>
22790 data"aaaalllllalaaalllalJfKg <gc>
22800 data 3567                   <gf>
22810 :                           <ca>
22820 data"JlCmaaaaaaaaaaaaaaaaaaa <nn>
22830 data"ammmmmmmmmmmmmmmaamma   <bb>
22840 data"ammmaaaaaaaaaaaaaBpNkJl
Caa                               <hc>
22850 data"ammmammmmmmmmmmmaammm   <oi>
22860 data"mamaaaaaaaaaaammJfKgCaJ
lCaa                              <hc>
22870 data"ammmmamamamamamaaaaama  <gf>
22880 data"ammmmmmmmmmmmmmmmmmma   <ig>
22890 data"aBmNlCaaaaaaaaaaaaaaaaa
a                                 <fh>
22900 data 3722                   <ok>
22910 :                           <oj>
22920 data"JiCnnaaBpNkCaaaaaaaaaaa
aaaa                              <al>
22930 data"JiCnnaaaaaanannannannaa <gk>
22940 data"aaanannannnnnanaanan   <gh>
22950 data"annnaannnBmNlJiCnnnanna
naa                               <oe>
22960 data"annnanaaaannnaaaanaa   <ai>
22970 data"aannanaannnnnnnnnnna   <am>
22980 data"aanaannannanaanaaana   <mf>
22990 data"naannannannannaanaaa   <kp>
23000 data 3733                   <gk>
23010 :                           <la>
23020 data"JfCaaaaaaaaoooooaaaaaoo <lj>
23030 data"aoooBpNkJfCooaooaaaaaa
oao                               <ik>
23040 data"aoaBmNlJfCoaoaaaoaaoao
ooa                               <lj>
23050 data"aooaoaoooooooooaoooa   <dg>
23060 data"aooaoaaaaaaaoooaooaa   <pg>
23070 data"aooaoooooooaaaooooao   <oo>
23080 data"aooaoooooaaoaooaaoao   <hk>
23090 data"aaaoaaaaaoooaooooaao   <ei>
23100 data 3621                   <bc>
23110 :                           <hl>
23120 data"BmCaaapaaBpNkBmCaaaapaa
aaaapp                            <pi>
23130 data"apapappppapappppapp    <jp>
23140 data"apapapaaapapaaaappap   <aa>
23150 data"apaaapapapaaappaapap   <do>
23160 data"BmNlBmCpapapapapppaappa
paa                               <le>
23170 data"apppppapappppaapappa   <bc>
23180 data"apapppappappppapapaa   <lm>
23190 data"paaaaaapppaaaaappaap   <ge>
23200 data 3701                   <if>
23210 :                           <ec>
23220 rem ======================= <og>
23230 rem 60671 bytes memory      <lj>
23240 rem 20205 bytes program     <ap>
23250 rem 00252 bytes variables   <le>
23260 rem 00040 bytes arrays      <hg>
23270 rem 00650 bytes strings     <on>
23280 rem 02048 bytes zeichensatz <ba>
23290 rem 37476 bytes free (0)    <nb>
23300 rem achtung !!!             <fd>
23310 rem datas ab zeile 20000    <dh>
23320 rem nicht umnummerieren.    <pm>
```

DISKSORTER PLUS

Disk-Jockey

Um eine Liste der Programme auf Ihren Disketten zu bekommen, brauchen Sie die Programmnamen nicht extra abzuschreiben, sondern nur zu übernehmen. Disksorter sorgt für Arbeitserleichterung und auf Wunsch auch für eine alphabetische Sortierung.

Das Programm Disksorter ist für den C16/116/Plus/4 geschrieben worden und dient zum Verwalten einer Liste aller Programme. Nach dem Laden und Starten mit RUN kann der Benutzer zwischen den zehn Menüpunkten 0 bis 9 wählen. Die Anwahl geschieht über die Cursor-hoch(-runter)-Taste und die Return-Taste. Es folgt eine Beschreibung, die jeden Menüpunkt genau erklärt, so daß es auch einem Anfänger gelingen wird, seine Programmnamen zu verwalten.

Punkt 0:
Mit Hilfe dieses Punktes können Sie die Programmnamen direkt aus dem Directory lesen. Nach der Anwahl werden Sie gebeten, die richtige Diskette ins Laufwerk zu legen und Return zu betätigen. Danach erscheinen in der unteren Bildschirmhälfte die Namen. Drücken Sie die Taste SPACE, so übernehmen Sie den Namen in Ihre Liste. Wenn Sie dies nicht wünschen, betätigen Sie die DEL-Taste. Eine besondere Funktion hat die ESC-Taste. Mit ihrer Hilfe werden Programme dem vorherigen zugeordnet, das heißt, die Blöcke aller zusammengehörigen Programme werden addiert und unter dem Namen des ersten ausgegeben.

Punkt 1:
Hier wird Ihre Liste auf dem Bildschirm dargestellt. Stoppt die Anzeige, so drücken Sie eine Taste.

Punkt 2:
Nach der Anwahl dieses Punktes wird Ihre Liste in alphabetischer Reihenfolge sortiert. Haben Sie etwas Geduld, es kann bei langen Listen eine Weile dauern.

Punkt 3:
Hier wird Ihnen die Möglichkeit geboten, Ihre Liste von Fehlern zu befreien. Mit den Cursor-Tasten (links/rechts) können Sie das gesuchte Element anwählen und mit SPACE verändern.
Mit DEL löschen Sie das Element aus der Liste und mit INST können Sie ein neues einfügen. Mit ESC gelangen Sie ins Hauptmenü.

Punkt 4:
Hier läßt sich eine vorher abgespeicherte Liste wieder in den Speicher laden und verarbeiten. Geben Sie den Listennamen an und drücken Sie die Return-Taste.

Punkt 5:
Haben Sie eine Liste erstellt, so können Sie sie auf Diskette abspeichern. Geben Sie den Namen ein, unter dem sie abgespeichert werden soll. Existiert dieser

Bitte lesen Sie weiter auf Seite 122

```
10 rem disksorter-plus=========c16 <gl>
20 rem (p) commodore welt team     <ho>
30 rem =========================== <ng>
40 rem (c) by bjoern schneider     <ml>
50 rem                             <pd>
60 rem basic v3.5                  <od>
70 rem c16/116/plus4               <ja>
80 rem floppy                      <ak>
90 rem =========================== <jg>
100 gosub 1570                     <fa>
110 color0,1:color4,1:color1,16,4:
scnclr                             <np>
120 dimd$(500):trap1450            <cg>
130 gosub1550:printcl$c4$"disksort
er-plus":fori=1to15:printzm$;:next <jf>
140 ifz=0thenu=0:elseu=z+1         <cl>
150 char,20,1,"("+str$(u)+" elemen
te )"+chr$(13)                     <ol>
160 printc4$c4$rn$" 0 "rf$"  neue
daten einlesen":printc4$" 1   date
n auf bildschirm"                  <jk>
170 printc4$" 2   daten sortieren"
:printc4$" 3   daten editieren"    <mf>
180 printc4$" 4   laden von daten"
:printc4$" 5   speichern von daten
"                                  <pp>
190 printc4$" 6   ausdrucken der l
iste":printc4$" 7   directory"     <eo>
200 printc4$" 8   liste von hand e
rgaenzen"                          <ch>
210 printc4$" 9   datei loeschen":
p=0:gosub1560:ifz<0thenz=0         <ep>
220 getkeya$:ifa$=chr$(13)then270  <pj>
230 char,0,p*2+4,str$(p)+" "       <he>
240 ifa$=c4$thenp=p+1:ifp=10thenp=
0                                  <if>
250 ifa$=c2$thenp=p-1:ifp=-1thenp=
9                                  <co>
260 char,0,p*2+4,rn$+str$(p)+" "+r
f$:goto220                         <bj>
270 onp+1goto290,590,690,790,1010,
1050,1110,1290,1330,1310           <cc>
280 rem *** neue datei einlesen **
*                                  <pg>
290 printcl$c4$c4$"richtige disket
te einlegen    "rn$"<return>"rf$   <ai>
300 getkeya$:ifa$<>chr$(13)then130 <nb>
310 printcl$c4$"bitte waehlen sie
:                  ";              <pa>
320 fori=1to19:printz2$;:next:prin
t                                  <dl>
330 printc4$c4$c4$"'space'   eleme
nt uebernehmen"                    <ma>
340 printc4$"'del'     element nic
ht nehmen"                         <pc>
350 printc4$"'esc'     zugehoerig
zum letzten"                       <fi>
360 printc4$c4$"  name
```

```
 id blk  nehmen?"                        <id>
370 fori=1to40:printzv$;:next:prin
tchr$(27)+"t"                            <ma>
380 printcl$;:ifz>0thenz=z+1             <mm>
390 rem directory lesen                  <je>
400 open15,8,15,"i0":open5,8,5,"#"
:gosub650:print#15,"u1 5 0 18 0"         <bc>
410 print#15,"b-p 5 162":get#5,a$:
get#5,b$:id$=a$+b$:t=18:s=1:fl=0         <lp>
420 print#15,"u1";5;0;t;s:get#5,a$
:t=asc(a$):get#5,a$:s=asc(a$)            <ni>
430 fori=0to7:an=32*i+2:print#15,"
b-p";5;an                                <lo>
440 q$="":foru=2to31:get#5,a$:ifa$
=""thena$=chr$(0)                        <eh>
450 q$=q$+a$:next:ifasc(q$)=0then5
30                                       <cl>
460 lo=asc(right$(q$,2)):hi=asc(ri
ght$(q$,1)):bl=lo+hi*256:x=bl:gosu
b660                                     <kd>
470 x$=mid$(q$,4,16)+" "+id$+" "+x
$                                        <dn>
480 print"  "x$"  ";                     <jj>
490 getkeya$:ifa$<>" "anda$<>chr$(
27)anda$<>chr$(20)then490                <kf>
500 ifa$=chr$(20)thenfl=0:printrn$
"nein"rf$:goto530                        <nn>
510 ifa$=chr$(27)then550                 <kl>
520 d$(z)=x$:z=z+1:m=z-1:fl=1:prin
trn$"ja"rf$"      "                      <nl>
530 nexti:ift>0then420                   <ej>
540 close5:close15:printhe$he$:z=z
-1:goto130                               <nj>
550 iffl=0then490:elseprintrn$"^^"
rf$                                      <fo>
560 b=val(right$(d$(m),3)):b=b+val
(right$(x$,3)):x=b:gosub660              <do>
570 d$(m)=left$(d$(m),len(d$(m))-3
)+x$:goto530                             <oo>
580 rem *** daten ausgeben ***           <ek>
590 printcl$;:t=0                        <ni>
600 i=0                                  <il>
610 printd$(t):i=i+1:t=t+1:ift-1=z
then640                                  <oa>
620 ifi=23thengetkeya$:goto600           <pk>
630 goto610                              <dd>
640 printc4$rn$"ende der liste"rf$
:getkeya$:goto130                        <hh>
650 ifds=0thenreturn:elsestop            <kd>
660 x$=str$(x):x$=right$(x$,len(x$
)-1):iflen(x$)>=3thenreturn              <db>
670 forut=1to3-len(x$):x$="0"+x$:n
ext:return                               <gh>
680 rem *** daten sortieren ***          <le>
690 printcl$c4$c4$"es sind"z+1" pr
ogramme gespeichert.":printc4$c4$c
4$"bitte warten !!!"                     <kp>
700 fori=1to1000:next:gosub1550          <fj>
710 l=z-1                                <fh>
720 forx=z-1to0step-1:f=0:fory=0to
l                                        <fg>
730 ifd$(y)<=d$(y+1)then750              <al>
740 f=y:a$=d$(y):d$(y)=d$(y+1):d$(
y+1)=a$                                  <hb>
750 next:l=f:iff=0then770                <io>
760 next                                 <hh>
770 gosub1560:goto130                    <ij>
780 rem *** editieren ***                <pe>
790 printcl$c4$"editieren der date
n :":printc4$c4$"right    ein elem
ent vor"                                 <pl>
800 printc4$"left      ein element
zurueck":printc4$"space    element
 edieren"                                <bm>
810 printc4$"esc       zurueck ins
menue":printc4$"del      element l
oeschen"                                 <fa>
820 printc4$"inst      element einf
uegen"                                   <cd>
830 printc4$c4$c4$"  name
    id blk"                              <if>
840 fori=1to40:printzv$;:next:prin
tchr$(27)+"t"                            <ab>
850 i=0                                  <gd>
860 printhe$c4$"  ";d$(i):getkeya$       <ef>
870 ifa$=c3$theni=i+1:ifi=z+1theni
=0                                       <gp>
880 ifa$=c1$theni=i-1:ifi=-1theni=
z                                        <pk>
890 ifa$=" "then940                      <nl>
900 ifa$=chr$(27)thenprinthe$+he$:
goto130                                  <pd>
910 ifa$=chr$(20)then950                 <kf>
920 ifa$=chr$(148)then970                <ag>
930 goto860                              <cd>
940 printc2$;:inputd$(i):goto860         <cd>
950 ifi=zthenz=z-1:i=0:goto860           <pj>
960 gosub1550:forr=i+1toz:d$(r-1)=
d$(r):next:z=z-1:gosub1560:goto860       <nl>
970 gosub1550:forr=ztoi+1step-1:d$
(r+1)=d$(r):next:d$(i+1)="
        -- ---"                          <el>
980 z=z+1:gosub1560                      <mp>
990 i=i+1:goto860                        <fj>
1000 rem *** laden von disk ***          <ha>
1010 printcl$c4$c4$"name der datei
 ??";:inputn$:open2,8,2,"0:"+n$+",
s,r":input#2,z                           <he>
1020 fori=0toz:input#2,d$(i):next:
printc4$c4$ds$                           <jm>
1030 close2:getkeya$:goto130             <bp>
1040 rem *** speichern ***               <ii>
1050 printcl$c4$c4$"name der zu sp
eichernen datei eingeben  ";:input
n$                                       <ch>
1060 printc4$c4$"soll datei uebers
```

```
chrieben werden (j/n)  ";                  <gj>
1070 inputj$:a$="":ifj$="j"thena$=
"§"                                        <mp>
1080 open1,8,1,a$+"0:"+n$+",s,w":p
rint#1,z:fori=0toz:print#1,d$(i):n
ext                                        <mn>
1090 close1:printc4$c4$ds$:getkeya
$:goto130                                  <hm>
1100 rem *** ausdrucken ***                <eb>
1110 ein=14:aus=15                         <kk>
1120 printcl$c4$c4$"name ??";:inpu
tn$:printc4$c4$"datum ??";:inputd$         <en>
1130 printcl$c4$c4$c4$"drucker fer
tig (j/n) ???":getkeya$:ifa$<>"j"t
hen130                                     <dm>
1140 open4,4:cmd4:print                    <jo>
1150 print"programmliste von ";chr
$(ein);n$;chr$(aus)"                       <bp>
1160 print"(aktuell vom ";d$;")":p
rintchr$(ein);"programme:";z+1;chr
$(aus):print                               <oi>
1170 a$=" name                 id blk
 ":a$=a$+a$+a$:printa$                     <od>
1180 a$="-------------------------
-":a$=a$+a$+a$:printa$                     <cn>
1190 f=int((((z+1)/3)+.1-int((z+1)
/3))*3)                                    <nl>
1200 iff=0thenq=0:elseq=1                  <gg>
1210 fori=0toint((z+1)/3)-1                <ke>
1220 print" "d$(i)"   "d$(int((z+1
)/3)+i+q)"   "d$(int((z+1)/3*2)+i+
q)                                         <bg>
1230 next                                  <ch>
1240 iff=0then1260                         <dd>
1250 print" "d$(i);:iff=2thenprint
"   "d$(i+i+1)                             <jl>
1260 print#4:close4                        <ip>
1270 goto130                               <el>
1280 rem *** directory ***                 <hh>
1290 scnclr:directory:getkeya$:got
o130                                       <ei>
1300 rem *** loeschen ***                  <ah>
1310 run                                   <na>
1320 rem *** ergaenzen ***                 <nb>
1330 ifz=0then1340:elsez=z+1               <ec>
1340 printcl$c4$c4$;z+1;c1$". elem
ent :":printc4$"(* eingeben, wenn
fertig!)"                                  <ga>
1350 printc4$c4$c4$"name     ??";:i
nputn$:iflen(n$)>16then1350                <oh>
1360 ifn$="*"thenz=z-1:goto130             <ci>
1370 printc4$"id         ??";:inputi$
:iflen(i$)<>2then1370                      <mm>
1380 printc4$"bloecke ??";:inputbl
:ifbl>999orbl<0then1380                    <kj>
1390 x=bl:gosub660:iflen(n$)=16the
n1410                                      <la>
1400 fori=1to16-len(n$):n$=n$+" ":
next                                       <lm>
1410 d$(z)=n$+" "+i$+" "+x$:printc
4$c4$d$(z):printc4$"richtig (j/n)
???"                                       <hm>
1420 getkeya$:ifa$="n"then1340             <fh>
1430 z=z+1:goto1340                        <gb>
1440 rem *** fehler-routine ***            <hn>
1450 close5:close15:close4:trap145
0:gosub1550                                <gg>
1460 printhe$he$cl$c4$c4$fl$" f e
h l e r  !!!"fo$                           <ll>
1470 printc4$rn$"diskettenstatus :
"rf$:printds$:printc4$rn$"basic-fe
hler :"rf$                                 <op>
1480 printerr$(er)" error in";elhe
n1400                                      <ek>
1490 printc4$c4$c4$rn$"bitte waehl
en :"rf$:printc4$"-1- weitermachen
"                                          <hn>
1500 printc4$"-2- zum menue":print
c4$"-3- ende":gosub1560                    <df>
1510 getkeya$                              <em>
1520 a=val(a$):ifa<1ora>3then1510          <hi>
1530 ifa=1thenresumenext                   <dj>
1540 ifa=2then130:elsescnclr:end           <og>
1550 poke65286,peek(65286)and239:r
eturn                                      <fa>
1560 poke65286,peek(65286)or16:ret
urn                                        <ck>
1570 rem nachspann ===============         <bf>
1580 rem * farbcodes/steuercodes *         <og>
1590 c4$=chr$(017):rn$=chr$(018)           <lg>
1600 he$=chr$(019):c3$=chr$(029)           <gp>
1610 fl$=chr$(130):fo$=chr$(132)           <bg>
1620 c2$=chr$(145):rf$=chr$(146)           <nh>
1630 cl$=chr$(147):c1$=chr$(157)           <ob>
1640 rem *** zeichensatz/graphik *         <hd>
1650 z2$=chr$(163):zm$=chr$(183)           <ih>
1660 zv$=chr$(192)                         <of>
1670 return                                <kh>
1680 rem =======================           <eo>
1690 rem 12277 bytes memory                <ad>
1700 rem 05728 bytes program               <bl>
1710 rem 00238 bytes variables             <ko>
1720 rem 01510 bytes arrays                <bf>
1730 rem 01371 bytes strings               <dp>
1740 rem 03430 bytes free (0)              <jf>
1750 rem =======================           <jp>
```

Disk-Jockey

Fortsetzung von Seite 120

Name schon auf Diskette, sollten Sie die nächste Frage mit J und RETURN quittieren, andernfalls tippen Sie N und RETURN.

Punkt 6:
Dies ist die Druckroutine. Mit ihrer Hilfe können Sie Ihre Liste zu Papier bringen, sofern Sie über einen

Drucker verfügen. Alle Modelle, die ich testen konnte, liefen einwandfrei, so daß es mit Ihrem Drucker auch keine Schwierigkeiten geben sollte.

Punkt 7:
Nach der Anwahl erscheint das Directory der eingelegten Diskette.

Punkt 8:
Bei diesem Menüpunkt haben Sie die Möglichkeit, Ihre Liste von Hand zu ergänzen. Sie müssen Name, die ID und die Anzahl der vom Programm belegten Blöcke eingeben. Sind Sie mit dem Ergänzen fertig, so tippen Sie anstatt des Namens das Zeichen „*“ ein, und Sie gelangen wieder ins Hauptmenü.

Punkt 9:
Dieser letzte Menüpunkt löscht den Speicher, und Sie können eine neue Liste laden oder eingeben und weiterarbeiten.

Björn Schneider □

EUROPE WAR

Taktik gegen Übermacht

Mit Geduld und strategischem Geschick erobern Sie die vom Feind besetzten Gebiete. Doch ist dies nicht ganz einfach, denn der Computer bekommt bisweilen Verstärkung.

Die Hälfte Europas befindet sich in der Hand von Superpower, der sich auch die restlichen Gebiete noch unter den Nagel reißen möchte. Ihre Aufgabe ist es, die Armeen Superpowers zu schlagen und alle Gebiete zurückzuerobern.
Der Angriff erfolgt durch Würfeln. Hat der Angreifer eine höhere Zahl, so verliert der Verteidiger eine Armee, ansonsten der Angreifer. Da bei Punktegleichheit der Verteidiger den Angriff abwehrt, ist er beim Würfeln etwas im Vorteil. Dies können Sie sich zunutze machen, um den wild angreifenden Gegner zu schwächen.
Zwischen zwei Gebieten ist in einer Runde nur zweimaliger Schlagabtausch möglich. Auch ist die Gesamtzahl der Angriffe pro Runde begrenzt. Sie haben im Gegensatz zu Superpower die Wahl, ob Sie angreifen wollen oder nicht. In jeder Runde bekommen die zwei Seiten zusätzliche Armeen, drei sind es zu Beginn. Je nach Anzahl der besetzten Gebiete ändert sich die Anzahl der dazukommenden Armeen.
Superpower bekommt bisweilen zusätzliche Armeen, mit denen er seine strategische Unterlegenheit nahezu ausgleichen kann. Besitzt er am Ende nur noch ein einziges Gebiet, das zudem nur von einer Seite aus angegriffen werden kann, ist es enorm schwierig, ihn dort wieder herauszulocken, um ihn völlig schlagen zu können. Einige Stunden müssen Sie sich Zeit nehmen, um Superpower zu besiegen. Doch aufgepaßt, nicht daß Sie am Ende der Besiegte sind. □

```
10 rem europe-war===============p4 <ak>
20 rem (p) commodore welt team     <ho>
30 rem ========================== <ng>
40 rem (c) by eckhard schulz       <no>
50 rem                             <pd>
60 rem                             <ah>
70 rem basic v3.5                  <nl>
80 rem plus4 (c16/116 + 64 kb)     <fd>
90 rem ========================== <jg>
100 gosub 3500                     <fc>
110 gosub3430:color0,1             <bg>
120 color4,1:printwh$              <hj>
130 gosub 2950                     <di>
140 printlg$he$left$(qd$,13)left$(
qr$,14)rn$"**************"         <ah>
150 printleft$(qr$,14)rn$"* europe
 war *                             <cj>
160 printleft$(qr$,14)rn$"********
******":gosub3190                  <eh>
170 printcl$c4$c4$c4$"die zeit ist
 vorbei----eine periode"           <ef>
180 printc4$"die in keinem geschic
htsbuch steht.":gosub3410          <jg>
190 printc4$c4$c4$"europa ist in 1
2 gebiete unterteilt."             <ko>
200 printc4$"und jedes gebiet hat
eine eigene armee."                <oj>
210 :gosub3410                     <mb>
220 printcl$c4$c4$c4$"da ist super
power,(ich),"                      <jn>
230 printc4$"ich versuche jedes ge
biet zu besetzen!"                 <pb>
240 gosub3410                      <nl>
250 printc4$c4$c4$"es ist deine au
fgabe mich zurueck-"               <be>
260 printc4$"zuschlagen und deine
gebiete, die                       <cj>
270 printc4$"bereits besetzt sind,
 wiederzubekommen.":gosub3410      <ij>
280 printcl$c4$c4$c4$"wir beide ha
ben am anfang 6 gebiete."          <mn>
290 printc4$"(meine sind mit einem
 '#' gezeichnet)"                  <bg>
300 printc4$"wir beide muessen uns
 schlagen"                         <jo>
310 printc4$"denn einer muss gewin
nen--und ich bin"                  <ga>
320 printc4$"bereit dich zu ueberl
aufen.":printc4$"also pass auf!!"  <ck>
330 printc4$c4$c4$rn$"druecke eine
 taste zum start.":gosub3190       <if>
340 dimp(12),c(12),fl(12),d(12),cd
(12),cr(12),hd(12),hr(12)          <gb>
350 gosub360:goto420               <mf>
360 n$="abcdefghjklm"              <kc>
370 ee$=" ":forx=1to39:e$=e$+ee$:e
q$=eq$+ee$:next:eq$=he$+c4$+eq$:e$
=he$+e$                            <ph>
```

```
380 forn=1to12:readcd(n),cr(n),hd(
n),hr(n):next                          <em>
390 xd$=c4$:forx=1to25:cd$=cd$+xd$
:next                                  <fk>
400 xr$=c3$:forx=1to39:cr$=cr$+xr$
:next                                  <oi>
410 return                             <mf>
420 printcl$c4$c4$c4$rn$"wuerfle w
er anfaengt."b3$"<taste>"              <gp>
430 p=rnd(rnd(0)):p=int((6*p)+1):g
osub3190                               <lo>
440 printc4$c4$c4$"dein wurf ";:pr
intp:gosub3190                         <af>
450 printc4$c4$"mein wurf ";           <bm>
460 c=rnd(rnd(0)):c=int((6*c)+1):i
fc=pthen460                            <ao>
470 printc:gosub3190:ifp>cthenprin
tc4$c4$c4$"du faengst an---":fl=1:
gosub3410                              <km>
480 ifp<cthenprintc4$c4$c4$"ich fa
nge an---":fl=2:gosub3410              <dh>
490 gosub2950:gosub1460:iffl=1then
620                                    <cp>
500 goto970                            <cc>
510 ml=0:hg=0:lm=0:gh=0:al=0:ab=0:
ba=0:bc=0:cd=0:dc=0:de=0:lj=0:lk=0
:ed=0                                  <bd>
520 jl=0:jk=0:jf=0:kl=0:kj=0:fe=0:
fj=0:ef=0:ge=0:gf=0:gj=0:gk=0:jg=0     <pj>
530 la=0:fg=0:eg=0:kg=0:return         <ed>
540 m=rnd(rnd(0)):m=int(15*m)          <cg>
550 ifm=1thenprinte$eq$:printhe$"5
 neue divisionen kommen extra!":go
sub3190                                <ma>
560 ifm=1theniffl=1thentp=5:goto61
0                                      <cg>
570 ifm=1thentc=5:goto610              <gg>
580 ifm=2thenprinte$eq$:printhe$"d
rei flugstaffeln sollen mir helfen
":gosub3190                            <ac>
590 ifm=2theniffl=1thentp=3:goto61
0                                      <ei>
600 ifm=2thentc=3                      <ej>
610 return                             <fg>
620 gosub540                           <le>
630 gosub510:fl=1:fora=1to12:ifp(a
)<>0thent1=t1+1                        <hh>
640 next:t1=int((t1/2)+.5)+tp:ift1
<3thent1=2                             <hn>
650 ift1<>0then830                     <mi>
660 ift1=0thenprinte$eq$:printhe$"
willst du angreifen (j/n)?"            <ke>
670 gosub3210:ifa$="n"thenprinte$:
goto970                                <lf>
680 ifa$<>"j"then660                   <hi>
690 printe$eq$:printhe$"angriff vo
n wo ?";:gosub3210:fw$="":fw$=a$:a
w=1                                    <jn>
700 printc3$rn$fw$:fora=1to300:nex
t:gosub920:ifaw=2then650               <mi>
710 aw=0:ifp(n)<2thenaa=1              <go>
720 ifaa=1thenprinte$eq$:printhe$"
du hast nicht genug armeen zum ang
riff"                                  <al>
730 ifaa=1thengosub3190:aa=0:goto6
50                                     <pn>
740 printe$eq$:printhe$"nach wo ?"
;:gosub3210:tw$="":tw$=a$:printc3$
rn$tw$                                 <fm>
750 gosub1610:iffw=1thenfw=0:goto6
50                                     <cc>
760 gosub1800:v=rnd(rnd(0)):v=int(
25*v)                                  <bc>
770 ifv=1thenprinte$eq$:printhe$"d
as muss ein panik-zug sein!!"          <mj>
780 ifv=2thenprinte$eq$:printhe$"d
u kannst es besser machen!!"           <gh>
790 ifv=3thenprinte$eq$:printhe$"e
in fuenkchen hoffnung im zug!!"        <cd>
800 fora=1to1500:next:gosub2600:fo
ra=1to1500:next:gosub2720:gosub325
0                                      <le>
810 pp=pp+1:ifpp=6then970              <ib>
820 goto650                            <co>
830 gosub860:p(n)=p(n)+1:t1=t1-1       <hd>
840 gosub1570                          <ck>
850 printhe$nd$nr$p(n):goto650         <el>
860 printe$eq$:printhe$rn$"druecke
 buchstaben um armeen zu setzen"rf
$                                      <lp>
870 print"du hast"t1"armeen zum se
tzen.":in=0                            <dk>
880 gosub3210                          <cd>
890 fora=1to12:ifmid$(n$,a,1)<>a$t
henin=in+1                             <fm>
900 next                               <jb>
910 ifin=12then860                     <fe>
920 n=0:fora=1to12:ifmid$(n$,a,1)=
a$andc(a)=0thenn=a                     <kd>
930 next:ifn<>0thenreturn              <ke>
940 printe$eq$:printhe$"das ist ni
cht dein gebiet!":gosub3190            <gh>
950 ifaw=1thenaw=2:return              <hj>
960 goto860                            <pn>
970 w=0:fl=2:b=0:t2=0:gosub510         <jl>
980 gosub540                           <ep>
990 fora=1to12:ifc(a)<>0thent2=t2+
1:b=b+1                                <ah>
1000 next:t2=int((t2/2)+.5)+tc:ift
2<3thent2=2                            <dc>
1010 printe$eq$:printhe$"nun bin i
ch dran!":gosub3190                    <mj>
1020 v=rnd(rnd(0)):v=int((v*3)+1):
ifv<>2then1080                         <ip>
1030 printe$eq$:ifb=11thenprinthe$
+"kaempfen bis zum ende ?"             <bj>
```

```
1040 ifb=9thenprinthe$+"ich fuehre
 nun!"                               <fb>
1050 ifb=6thenprinthe$+"wir sind b
eide gleich,aber nicht lange!"       <kj>
1060 ifb=3thenprinthe$+"ich habe n
och ein bischen aufzuholen!"         <gg>
1070 ifb=1thenprinthe$+"meine letz
te festung!!"                        <ek>
1080 gosub3190:printe$eq$:printhe$
"ich setze"t2"armeen nach--":gosub
3410                                 <oi>
1090 goto1110                        <da>
1100 t2=t2-1:gosub2570:ift2=0then2
110                                  <ca>
1110 ifc(12)>0and(c(12)-p(11))<0th
enc(12)=c(12)+1:n2=12:goto1100       <bk>
1120 b=c(11):ifb>0and(b-p(12))<0th
enc(11)=c(11)+1:n2=11:goto1100       <mi>
1130 ifb>0and(b-p(1))<0thenc(11)=c
(11)+1:n2=11:goto1100                <dc>
1140 ifb>0and(b-p(9))<0thenc(11)=c
(11)+1:n2=11:goto1100                <di>
1150 ifb>0and(b-p(10))<0thenc(11)=
c(11)+1:n2=11:goto1100               <ng>
1160 b=c(1):ifb>0and(b-p(11))<0the
nc(1)=c(1)+1:n2=1:goto1100           <kd>
1170 ifb>0and(b-p(2))<0thenc(1)=c(
1)+1:n2=1:goto1100                   <ae>
1180 b=c(2):ifb>2and(b-p(1))<0then
c(2)=c(2)+1:n2=2:goto1100            <ei>
1190 ifb>0and(b-p(3))<0thenc(2)=c(
2)+1:n2=2:goto1100                   <ci>
1200 b=c(3):ifb>0and(b-p(2))<0then
c(3)=c(3)+1:n2=3:goto1100            <hl>
1210 ifb>0and(b-p(4))<0thenc(3)=c(
3)+1:n2=3:goto1100                   <jh>
1220 b=c(4):ifb>0and(b-p(3))<0then
c(4)=c(4)+1:n2=4:goto1100            <jk>
1230 ifb>0and(b-p(5))<0thenc(4)=c(
4)+1:n2=4:goto1100                   <ef>
1240 b=c(5):ifb>0and(b-p(4))<0then
c(5)=c(5)+1:n2=5:goto1100            <hh>
1250 ifb>0and(b-p(6))<0thenc(5)=c(
5)+1:n2=5:goto1100                   <dm>
1260 ifb>0and(b-p(7))<0thenc(5)=c(
5)+1:n2=5:goto1100                   <lj>
1270 b=c(9):ifb>0and(b-p(11))<0the
nc(9)=c(9)+1:n2=9:goto1100           <be>
1280 ifb>0and(b-p(6))<0thenc(9)=c(
9)+1:n2=9:goto1100                   <lg>
1290 ifb>0and(b-p(10))<0thenc(9)=c
(9)+1:n2=9:goto1100                  <nj>
1300 ifb>0and(b-p(7))<0thenc(9)=c(
9)+1:n2=9:goto1100                   <jc>
1310 b=c(10):ifb>0and(b-p(11))<0th
enc(10)=c(10)+1:n2=10:goto1100       <ho>
1320 ifb>0and(b-p(9))<0thenc(10)=c
(10)+1:n2=10:goto1100                <nm>
1330 ifb>0and(b-p(7))<0thenc(10)=c
(10)+1:n2=10:goto1100                <dl>
1340 b=c(6):ifb>0and(b-p(5))<0then
c(6)=c(6)+1:n2=6:goto1100            <na>
1350 ifb>0and(b-p(9))<0thenc(6)=c(
6)+1:n2=6:goto1100                   <eh>
1360 ifb>0and(b-p(7))<0thenc(6)=c(
6)+1:n2=6:goto1100                   <cm>
1370 ifc(8)>0and(c(8)-p(7))<0thenc
(8)=c(8)+1:n2=8:goto1100             <ni>
1380 b=c(7):ifb>0and(b-p(8))<0then
c(7)=c(7)+1:n2=7:goto1100            <ho>
1390 ifb>0and(b-p(5))<0thenc(7)=c(
7)+1:n2=7:goto1100                   <id>
1400 ifb>0and(b-p(9))<0thenc(7)=c(
7)+1:n2=7:goto1100                   <im>
1410 ifb>0and(b-p(10))<0thenc(7)=c
(7)+1:n2=7:goto1100                  <na>
1420 ifb>0and(b-p(6))<0thenc(7)=c(
7)+1:n2=7:goto1100                   <di>
1430 v=rnd(rnd(0)):v=int((12*v)+1)
:ifc(v)=0then1430                    <fi>
1440 c(v)=c(v)+1:n2=v:goto1100       <pj>
1450 gosub3250:goto620               <mn>
1460 fora=1to12                      <am>
1470 v=rnd(rnd(0)):v=int((12*v)+1)
:iffl(v)<>0then1470                  <aa>
1480 ifa>6thenc(v)=5:goto1500        <mj>
1490 p(v)=5                          <gc>
1500 fl(v)=1:next                    <lp>
1510 fori=1to12:n=i                  <bh>
1520 ifp(i)=0then1540                <ac>
1530 gosub1570:printhe$nd$nr$p(i):
goto1550                             <be>
1540 gosub1570:printhe$nd$nr$c(i):
printhe$hd$hr$"#"                    <ol>
1550 next                            <kk>
1560 return                          <mk>
1570 nd$="":nr$="":hd$="":hr$=""     <ib>
1580 nd$=left$(cd$,cd(n)):nr$=left
$(cr$,cr(n))                         <mo>
1590 hd$=left$(cd$,hd(n)):hr$=left
$(cr$,hr(n))                         <hk>
1600 return                          <bk>
1610 n=0:fora=1to12:ifmid$(n$,a,1)
=tw$andc(a)<>0thenn=a                <eo>
1620 next:ifn<>0then1650             <oo>
1630 printe$eq$:printhe$"das ist s
chon dein gebiet,dummkopf!!":gosub
3190:fw=1                            <gp>
1640 return                          <gk>
1650 iftw$="a"and(fw$="b"orfw$="l"
)then1790                            <nm>
1660 iftw$="b"and(fw$="a"orfw$="c"
)then1790                            <kj>
1670 iftw$="c"and(fw$="b"orfw$="d"
)then1790                            <fj>
1680 iftw$="d"and(fw$="c"orfw$="e"
```

```
)then1790                               <an>
1690 iftw$="e"and(fw$="d"orfw$="f"
orfw$="g")then1790                      <mh>
1700 iftw$="f"and(fw$="e"orfw$="g"
orfw$="j")then1790                      <me>
1710 iftw$="g"and(fw$="e"orfw$="f"
orfw$="j"orfw$="k"orfw$="h")then17
90                                      <ab>
1720 iftw$="h"andfw$="g"then1790        <gh>
1730 iftw$="j"and(fw$="f"orfw$="g"
orfw$="k"orfw$="l")then1790             <pl>
1740 iftw$="k"and(fw$="g"orfw$="j"
orfw$="l")then1790                      <be>
1750 iftw$="l"and(fw$="k"orfw$="j"
orfw$="a"orfw$="m")then1790             <pe>
1760 iftw$="m"andfw$="l"then1790        <oh>
1770 printe$eq$:printhe$"du kannst
 "rn$fw$rf$" nicht angreifen ":fw=
1                                       <ga>
1780 gosub3190:goto1790                 <oo>
1790 return                             <jg>
1800 iffw$="m"and(tw$="l")thenml=m
l+1:ifml>2then2100                      <pe>
1810 iffw$="h"and(tw$="g")thenhg=h
g+1:ifhg>2then2100                      <jb>
1820 iffw$="l"and(tw$="m")thenlm=l
m+1:iflm>2then2100                      <in>
1830 iffw$="g"and(tw$="h")thengh=g
h+1:ifgh>2then2100                      <ip>
1840 iffw$="a"and(tw$="l")thenal=a
l+1:ifal>2then2100                      <ei>
1850 iffw$="b"and(tw$="a")thenba=b
a+1:ifba>2then2100                      <gd>
1860 iffw$="b"and(tw$="c")thenbc=b
c+1:ifbc>2then2100                      <kc>
1870 iffw$="c"and(tw$="d")thencd=c
d+1:ifcd>2then2100                      <cb>
1880 iffw$="d"and(tw$="c")thendc=d
c+1:ifdc>2then2100                      <cp>
1890 iffw$="d"and(tw$="e")thende=d
e+1:ifde>2then2100                      <lm>
1900 iffw$="l"and(tw$="j")thenlj=l
j+1:iflj>2then2100                      <dp>
1910 iffw$="l"and(tw$="k")thenlk=l
k+1:iflk>2then2100                      <ef>
1920 iffw$="e"and(tw$="d")thened=e
d+1:ifed>2then2100                      <hk>
1930 iffw$="j"and(tw$="l")thenjl=j
l+1:ifjl>2then2100                      <kb>
1940 iffw$="j"and(tw$="k")thenjk=j
k+1:ifjk>2then2100                      <om>
1950 iffw$="j"and(tw$="f")thenjf=j
f+1:ifjf>2then2100                      <dk>
1960 iffw$="k"and(tw$="l")thenkl=k
l+1:ifkl>2then2100                      <ne>
1970 iffw$="k"and(tw$="j")thenkj=k
j+1:ifkj>2then2100                      <an>
1980 iffw$="f"and(tw$="e")thenfe=f
e+1:iffe>2then2100                      <mm>
1990 iffw$="f"and(tw$="j")thenfj=f
j+1:iffj>2then2100                      <nl>
2000 iffw$="e"and(tw$="f")thenef=e
f+1:ifef>2then2100                      <bo>
2010 iffw$="g"and(tw$="e")thenge=g
e+1:ifge>2then2100                      <eb>
2020 iffw$="g"and(tw$="f")thengf=g
f+1:ifgf>2then2100                      <hf>
2030 iffw$="g"and(tw$="j")thengj=g
j+1:ifgj>2then2100                      <nn>
2040 iffw$="g"and(tw$="k")thengk=g
k+1:ifgk>2then2100                      <ka>
2050 iffw$="j"and(tw$="g")thenjg=j
g+1:ifjg>2then2100                      <of>
2060 iffw$="f"and(tw$="g")thenfg=f
g+1:iffg>2then2100                      <ke>
2070 iffw$="e"and(tw$="g")theneg=e
g+1:ifeg>2then2100                      <fk>
2080 iffw$="k"and(tw$="g")thenkg=k
g+1:ifkg>2then2100                      <ce>
2090 return                             <pc>
2100 printe$eq$:printhe$"nimm eine
n anderen angriff!":gosub3190:goto
650                                     <in>
2110 fora=1to12:y=c(a):poke832+a,y
:y=p(a):poke832+a+12,y:next             <gg>
2120 clr:gosub3500:dimp(12),c(12),
fl(12),d(12),cd(12),cr(12),hd(12),
hr(12)                                  <ne>
2130 fora=1to12:d=peek(832+a):c(a)
=d:d=peek(832+a+12):p(a)=d:next         <dj>
2140 gosub360:goto2170                  <ij>
2150 cc=cc+1:ifcc=6then620              <mn>
2160 fl=2:gosub2490                     <bo>
2170 ifml<2thenifc(12)>2andp(11)<>
0thenfw$="m":tw$="l":ml=ml+1:goto2
150                                     <li>
2180 ifhg<2thenifc(8)>2andp(7)<>0t
henfw$="h":tw$="g":hg=hg+1:goto215
0                                       <ll>
2190 iflm<2thenifc(11)>2andp(12)<>
0thenfw$="l":tw$="m":lm=lm+1:goto2
150                                     <le>
2200 ifgh<2thenifc(7)>2andp(8)<>0t
henfw$="g":tw$="h":gh=gh+1:goto215
0                                       <kd>
2210 ifal<2thenifc(1)>2andp(11)<>0
thenfw$="a":tw$="l":al=al+1:goto21
50                                      <fh>
2220 ifab<2thenifc(1)>2andp(2)<>0t
henfw$="a":tw$="b":ab=ab+1:goto215
0                                       <jk>
2230 ifba<2thenifc(2)>2andp(1)<>0t
henfw$="b":tw$="a":ba=ba+1:goto215
0                                       <on>
2240 ifbc<2thenifc(2)>2andp(3)<>0t
henfw$="b":tw$="c":bc=bc+1:goto215
```

```
0                                        <lb>
2250 ifcd<2thenifc(3)>2andp(4)<>0t
henfw$="c":tw$="d":cd=cd+1:goto215
0                                        <aa>
2260 ifdc<2thenifc(4)>2andp(3)<>0t
henfw$="d":tw$="c":dc=dc+1:goto215
0                                        <ej>
2270 ifde<2thenifc(4)>2andp(5)<>0t
henfw$="d":tw$="e":de=de+1:goto215
0                                        <kd>
2280 ifla<2thenifc(11)>2andp(1)<>0
thenfw$="l":tw$="a":la=la+1:goto21
50                                       <pp>
2290 iflj<2thenifc(11)>2andp(9)<>0
thenfw$="l":tw$="j":lj=lj+1:goto21
50                                       <bd>
2300 iflk<2thenifc(11)>2andp(10)<>
0thenfw$="l":tw$="k":lk=lk+1:goto2
150                                      <pg>
2310 ifed<2thenifc(5)>2andp(4)<>0t
henfw$="e":tw$="d":ed=ed+1:goto215
0                                        <ga>
2320 ifjl<2thenifc(9)>2andp(11)<>0
thenfw$="j":tw$="l":jl=jl+1:goto21
50                                       <jg>
2330 ifjk<2thenifc(9)>2andp(10)<>0
thenfw$="j":tw$="k":jk=jk+1:goto21
50                                       <fb>
2340 ifjf<2thenifc(9)>2andp(6)<>0t
henfw$="j":tw$="f":jf=jf+1:goto215
0                                        <ad>
2350 ifkl<2thenifc(10)>2andp(11)<>
0thenfw$="k":tw$="l":kl=kl+1:goto2
150                                      <ij>
2360 ifkj<2thenifc(10)>2andp(9)<>0
thenfw$="k":tw$="j":kj=kj+1:goto21
50                                       <he>
2370 iffe<2thenifc(6)>2andp(5)<>0t
henfw$="f":tw$="e":fe=fe+1:goto215
0                                        <ic>
2380 iffj<2thenifc(6)>2andp(9)<>0t
henfw$="f":tw$="j":fj=fj+1:goto215
0                                        <eb>
2390 ifef<2thenifc(5)>2andp(6)<>0t
henfw$="e":tw$="f":ef=ef+1:goto215
0                                        <hf>
2400 ifge<2thenifc(7)>2andp(5)<>0t
henfw$="g":tw$="e":ge=ge+1:goto215
0                                        <hj>
2410 ifgf<2thenifc(7)>2andp(6)<>0t
henfw$="g":tw$="f":gf=gf+1:goto215
0                                        <bi>
2420 ifgj<2thenifc(7)>2andp(9)<>0t
henfw$="g":tw$="j":gj=gj+1:goto215
0                                        <mj>
2430 ifgk<2thenifc(7)>2andp(10)<>0
thenfw$="g":tw$="k":gk=gk+1:goto21
50                                       <go>
2440 ifjg<2thenifc(9)>2andp(7)<>0t
henfw$="j":tw$="g":jg=jg+1:goto215
0                                        <cb>
2450 iffg<2thenifc(6)>2andp(7)<>0t
henfw$="f":tw$="g":fg=fg+1:goto215
0                                        <ep>
2460 ifeg<2thenifc(5)>2andp(7)<>0t
henfw$="e":tw$="g":eg=eg+1:goto215
0                                        <kf>
2470 ifkg<2thenifc(10)>2andp(7)<>0
thenfw$="k":tw$="g":kg=kg+1:goto21
50                                       <nf>
2480 goto620                             <fb>
2490 gosub3190:printe$eq$:printhe$
"ich greife an von "rn$fw$rf$" nac
h "rn$tw$                                <li>
2500 gosub3190:gosub2600:fora=1to2
000:next:gosub2720:gosub3250             <in>
2510 ifp(n2)<1then2530                   <fk>
2520 goto2170                            <lj>
2530 v=rnd(rnd(0)):v=int((9*v)+1)        <fn>
2540 ifc(n1)-v<2then2530                 <ga>
2550 c(n1)=c(n1)-v:c(n2)=c(n2)+v         <ik>
2560 n=n1:gosub1570:printhe$nd$nr$
c(n)                                     <ld>
2570 n=n2:gosub1570:printhe$nd$nr$
rn$c(n):gosub3190                        <jd>
2580 n=n2:gosub1570:printhe$nd$nr$
c(n)                                     <hl>
2590 gosub1570:printhe$hd$hr$"#":g
osub3250:return                          <if>
2600 v=rnd(rnd(0)):v=int((6*v)+1)        <ki>
2610 iffl=1thenp=v:goto2630              <ii>
2620 c=v                                 <pi>
2630 v=rnd(rnd(0)):v=int((6*v)+1)        <el>
2640 iffl=1thenc=v:gosub2700:goto2
680                                      <ai>
2650 p=v:gosub2700                       <mn>
2660 ifp=>cthenprinte$eq$:printhe$
"du hast meinen angriff abgewehrt!
":return                                 <ao>
2670 printe$eq$:printhe$"mein angr
iff war erfolgreich!":return             <jh>
2680 ifc=>pthenprinte$eq$:printhe$
"ich habe deinen angriff abgewehrt
!":return                                <jl>
2690 printe$eq$:printhe$"dein angr
iff war erfolgreich!":return             <po>
2700 printe$eq$:printhe$"dein wurf
 ";:gosub3410:printp;:gosub3190          <gd>
2710 printb3$"mein wurf ";:gosub34
10:printc:gosub3190:return               <ga>
2720 fora=1to12:ifmid$(n$,a,1)=fw$
thenn1=a                                 <hp>
2730 ifmid$(n$,a,1)=tw$thenn2=a          <lb>
2740 next                                <pp>
2750 iffl=1andp>cthenc(n2)=c(n2)-1       <gf>
2760 iffl=1andc=>pthenp(n1)=p(n1)-
```

```
1                                   <hp>
2770 iffl=2andp=>cthenc(n1)=c(n1)-
1                                   <io>
2780 iffl=2andc>pthenp(n2)=p(n2)-1  <ni>
2790 iffl=1thenn=n1:goto2810        <pc>
2800 n=n2                           <kp>
2810 gosub1570:printhe$nd$nr$b3$:p
rinthe$nd$nr$p(n)                   <oa>
2820 iffl=1thenn=n2:goto2840        <ap>
2830 n=n1                           <jm>
2840 gosub1570:printhe$nd$nr$b3$:p
rinthe$nd$nr$c(n):iffl=1then2860    <dl>
2850 return                         <oi>
2860 ifc(n)>=1thenreturn            <cj>
2870 printe$eq$:printhe$"anzahl de
r besetzenden armeen (1-9)?":gosub
3210                                <ea>
2880 v=val(a$):e=0:ford=1to9:ifv=d
thene=e+1                           <jh>
2890 next:ife=0then2860             <fp>
2900 ifp(n1)-v<1then2860            <ia>
2910 p(n1)=p(n1)-v:p(n2)=p(n2)+v    <lj>
2920 n=n1:gosub1570:printhe$nd$nr$
p(n)c1$" "                          <nj>
2930 n=n2:gosub1570:printhe$nd$nr$
p(n)                                <in>
2940 gosub1570:printhe$hd$hr$" ":r
eturn                               <fg>
2950 poke804,57:poke805,1           <dj>
2960 print"JdBbbbKeeeeeeeeeeeeeee
eeeeeeeeeeeeeeeeeeeeee"             <lf>
2970 print"BcCaaaaaaaaaaaaaaaaaaJ
cCaaaMoCaaaaMoCaEeCaaKfCaaaaaKh"    <dk>
2980 print"BcCaaaaaaaaaaaaaaaaaJc
KjCaaaKfEdKhBcCaaJcKfCaaaMoCaaaaaa
Kh"                                 <kj>
2990 print"BcCaaaaaaaaaaaKmLjCkkkk
kJcKfCaaKefCaaNpBcCaNpJcKeeMoCaaaa
aaaKh"                              <oo>
3000 print"BcCaEbFeEmboFeEjlCaKmJc
CaLmBcLlCaaaaJcKfEcBcKjNpJcCaaa";   <jp>
3010 print"BcKjCaaaJcKjCaaaaaEfCaa
Kh"                                 <ld>
3020 print"BcCaaaaaaaaKeemJcCaBcLo
CaaaaaKddCaKhJcCaaKhBcCaaaKeJcKfCa
aaaaaaaKh"                          <ck>
3030 print"BcCaaaaaaKhJcKdCaKhBcLg
JcCaLgBcCaEopFcEeCnaaKhJcCaaBcKjCa
aJcKjCaaaaaaaaaaKh"                 <bg>
3040 print"BcCaaaaaaJcCaaLkBcKmLjJ
cEbKhBcCaFdEffCaKhJcKdNpBcCaNpKjCa
aaNpJcCaaaaaaaaaaKh"                <bm>
3050 print"BcCaaaaaaLiKdCaLjJcCaaa
LgBcCaaKeeJcKjCaaKdNpBcKeJcKjdddMn
KeeCaaaaaaaKh"                      <ll>
3060 print"BcCaaaaaaaaKmJcLoCaaaBc
LlKeJcKjCaaaaaaKhCaaaaaaaaKfCaaaaa
aKh"                                <km>
3070 print"BcCaaaaaaaaaaakaJcKjMnC
aaaEkCaaaMoCaaaEgCaaaaKfCaaaaaaKh" <be>
3080 print"BcCaaaaaaaaNpJcKddddCaa
MnCaaaaaMoCaaaaaaaaMoCaaaaaaaKh"    <me>
3090 print"BcCaaaaaaaaaNpJcCaaEmCa
aKhCaaaaaMnKeeeCaaaaKheeeCaaaaaKh" <od>
3100 print"BcCaaaaaaaaaKhJcCaaaaaM
oKeeeeeMoCaaaMnKeeeMoCaaKhCaaaaaKh
"                                   <ga>
3110 print"BcCaaKhJcKdddNpBcKeeJcK
jCaaaaaKdMoCaaaaKfCaaaaaaaaaaaMnBc
Kjdddd"                             <ho>
3120 print"BcCaaJcKjCaEnCaaaaMnCaa
aaaaMnCaaaNpBcKdNpJcCaaaaaEhCaaaaa
BcFdEdiFhCo"                        <jk>
3130 print"BcCaJcKjCaaaaaaaaKdMnBc
KfCaaaaaNpJcCaaNpBcCaNpJcCaaaaaaaa
aaBcEnffFcCa"                       <nc>
3140 print"BcCaJcCaaaaaaaaaBcKjCaa
aaaaaaNpJcCaElNpBcCaaNpJcCaaaaaaaa
NpBcCaaJcKjh"                       <bh>
3150 print"BcCaJcCaaaaaaaBcKjCaaaE
njFeeEfmCaaaNpJcCaaNpBcCaaJcCaaaaa
aMoKdCaaEiKh"                       <ol>
3160 print"BcCaaNpJcCaaaaBcKjCaaaa
aEnffFcCaaaaaaNpCaLoCaNpJcCaNpBcCa
aaNpJcCaaaaKh"                      <fi>
3170 print"BcCaaaaaaaaaaaaaaaaaaaa
aaaLmCaaaaNpJcCaNpBcCaaaNpJcKeeeLk
"                                   <pp>
3180 poke804,75:poke805,236:return <jo>
3190 poke239,0:wait239,1            <no>
3200 return                         <kh>
3210 geta$:ifa$=""then3210          <mf>
3220 return                         <mp>
3230 data11,10,9,11,5,19,6,19,7,22
,8,23,5,28,4,30,9,32,10,32,14,24,1
5,27,20,27                          <ej>
3240 data18,26,22,36,22,34,14,17,1
3,16,18,17,17,18,17,11,15,14,20,3,
18,6                                <mh>
3250 s=0:t=0:fora=1to12:ifc(a)<>0t
hens=s+1                            <fe>
3260 ifp(a)<>0thent=t+1             <ki>
3270 next:ifs=12then3300            <nj>
3280 ift=12then3340                 <hb>
3290 return                         <fl>
3300 printcl$c4$c4$"aha! ich habe
gewonnen--":gosub3410               <gb>
3310 printc4$c4$"meine invasions-s
trategie war brilliant":printc4$"w
ie immer!!"                         <id>
3320 gosub3410:printc4$c4$rn$"niem
and"rf$" kann mich stoppen!":gosub
3410                                <pg>
3330 printc4$rn$"heute europa----m
orgen die ganze welt!!!":gosub3410
:goto3380                           <be>
```

```
3340 printcl$c4$c4$"bahl-du hast g
ewonnen":gosub3410                <kj>
3350 printc4$"das auch nur durch e
inen fehler"                      <lc>
3360 printc4$"in meiner konzentrat
ion!":gosub3410                   <fo>
3370 printc4$c4$"es passiert besti
mmt nicht wieder!":gosub3410      <me>
3380 printc4$c4$"willst du noch ei
nmal spielen (j/n)?"              <fl>
3390 gosub3210:ifa$<>"j"then3390  <fc>
3400 run                          <de>
3410 fortq=1to3000:next           <md>
3420 return                       <ga>
3430 restore3450:fori=313to341    <na>
3440 reada:pokei,a:next:restore:re
turn                              <ef>
3450 data 201,013,240,022,201,193 <mg>
3460 data 048,012,056,233,193,010 <fm>
3470 data 010,010,010,141,056,001 <pf>
3480 data 024,096,056,233,065,013 <ij>
3490 data 056,001,076,075,236     <mk>
3500 rem nachspann =============== <hc>
3510 rem * farbcodes/steuercodes * <gf>
3520 wh$=chr$(005):c4$=chr$(017)  <al>
3530 rn$=chr$(018):he$=chr$(019)  <lc>
3540 c3$=chr$(029):rf$=chr$(146)  <lp>
3550 cl$=chr$(147):lg$=chr$(153)  <op>
3560 c1$=chr$(157)                <fd>
3570 rem ********* zeichenfolgen * <nb>
3580 for q=1 to 40                <nk>
3590 qd$=qd$+c4$:qr$=qr$+c3$      <lh>
3600 nextq:b$=chr$(32)            <bp>
3610 b3$=b$+b$+b$:return          <lp>
3620 rem ======================== <jo>
3630 rem 60671 bytes memory       <pb>
3640 rem 14124 bytes program      <hb>
3650 rem 00245 bytes variables    <dp>
3660 rem 00576 bytes arrays       <ia>
3670 rem 00616 bytes strings      <lp>
3680 rem 45110 bytes free (0)     <ac>
3690 rem ======================== <bi>
```


VIER WÜRFEL

Wer schafft's

Vier nebeneinander liegende farbige Würfel sind so zu drehen, daß Oberseiten, Unterseiten, Vorderseiten und Hinterseiten je vier unterschiedliche Farben aufweisen.

Vier Würfel ist ein Knobelspiel. Das Bild zeigt vier Würfel. Da es nicht möglich ist, alle sechs Seiten eines Würfels gleichzeitig zu sehen, erscheinen die nicht sichtbaren Seiten unterhalb der Würfel. Sie müssen sich vorstellen, daß die Rückseite sowie die linke und untere Seite nach unten gezogen wurden, so daß Sie in das Innere des Würfels blicken.

Die 24 Seiten der Würfel sind in vier verschiedenen Farben eingefärbt, jedoch völlig unregelmäßig. Es ist durchaus möglich, daß eine Farbe auf einem Würfel drei- bis viermal vorhanden ist, auf einem anderen aber fehlt.

Unter dem ersten Würfel erscheint ein blinkender Pfeil. Durch Druck auf den Feuerknopf wandert dieser Pfeil von Würfel zu Würfel. Der Würfel, auf den der Pfeil zeigt, kann in alle sechs Richtungen gedreht werden. Dies geschieht ebenfalls mit Hilfe des Joysticks. Dabei gilt folgende Regel:

- oben = kippen nach hinten;
- unten = kippen nach vorne;
- rechts = drehen nach rechts;
- links = drehen nach links;
- oben und rechts = kippen nach rechts;
- oben und links = kippen nach links.

Das Ziel des Spieles ist es, die Würfel so zu drehen, daß die Reihen oben, hinten, unten und vorne keine Farbe mehrmals zeigen. Das heißt also, daß alle vier Farben in den genannten Reihen vertreten sein müssen. Die rechte und linke Seite der einzelnen Würfel ist dabei ohne Bedeutung.

Zur besseren Übersicht werden die Farben der genannten Reihen unten rechts noch einmal angezeigt und die Reihen, die schon richtig sortiert sind, blinken. Wenn alle vier Reihen blinken, ist das Ziel des Spieles erreicht.

Danach werden Sie gefragt, ob Sie Schluß machen wollen. Bei Eingabe von J ist das Spiel zu Ende. Geben Sie N ein, können Sie sich entscheiden, ob Sie das Standardspiel spielen wollen oder ob Sie es vorziehen, die Würfelfarben durch den Zufall bestimmen zu lassen. Gleich, ob Standard- oder Zufallspiel, es wird immer gewährleistet, daß das Spiel eine Lösung hat. Vielleicht sollten Sie zuerst eine Weile mit dem Standardspiel üben, weil dann die Ausgangsbedingungen immer gleich sind.

Sollten Sie sich einmal völlig verfranzt haben oder keine Lust mehr verspüren, so bringt die RETURN-Taste wieder das Auswahlbild zurück.

In der Zeile oberhalb der Würfel wird ganz links die Anzahl der Versuche und über jedem Würfel die Anzahl der Drehungen angezeigt.

Das Spiel ist auf dem Plus/4 und auf dem C16/116 lauffähig.

G. Kramer □

VIER WUERFEL

```
10 rem vier wuerfel=========c16/p4 <pj>
20 rem (p) commodore welt        == <hf>
30 rem ========================== <ng>
40 rem (c) by g. kramer          == <eg>
50 rem                           == <if>
60 rem                           == <nd>
70 rem basic 3.5   40z/ascii     == <ef>
80 rem c16/p4 + 1531/1551/41/71  == <hb>
90 rem ========================== <jg>
100 gosub3030                       <ee>
110 gosub230                        <kg>
120 gosub1280                       <mc>
130 gosub2450                       <cf>
140 ifa$="a"then110                 <fb>
150 ifa$="j"thenend                 <id>
160 gosub2700                       <ol>
170 goto110                         <em>
180 getkeya$                        <nk>
190 end                             <pn>
200 ************************        <hk>
210 *** wuerfel zeichnen ***        <gl>
220 ************************        <gg>
230 a$=chr$(160)+chr$(160)          <jh>
240 a0$=chr$(233)+a$+a$+chr$(105)   <je>
250 a$=a$+a$+chr$(160)              <da>
260 a1$=chr$(90)+chr$(90)           <kc>
270 a1$=chr$(90)+a1$+a1$+chr$(124)  <kb>
280 fori=0to3                       <nd>
290 a=b+81+10*i                     <fe>
300 fori1=0to1                      <nj>
310 a=a+i1*39                       <go>
320 fori0=1tolen(a0$)               <oh>
330 pokea+i0,asc(mid$(a0$,i0,1))    <pf>
340 next                            <cm>
350 pokea+len(a0$)+1,255-165*i1     <bj>
360 next                            <fe>
370 pokea+len(a0$)+1,255            <lb>
380 pokea+len(a0$)+2,233            <ag>
390 a=a+40                          <gm>
400 fori1=1tolen(a1$)               <pk>
410 pokea+i1,asc(mid$(a1$,i1,1))    <hp>
420 next                            <mn>
430 pokea+len(a1$)+1,233            <dc>
440 pokea+len(a1$)+2,160            <pa>
450 fori0=0to4                      <pf>
460 a=a+40                          <pl>
470 fori1=1tolen(a$)                <kc>
480 pokea+i1,asc(mid$(a$,i1,1))     <ph>
490 next                            <fj>
500 ifi0=4then560                   <no>
510 pokea+len(a$)+1,90              <op>
520 pokea+len(a$)+2,160             <go>
530 ifi0=3then580                   <jg>
540 pokea+len(a$)+3,160             <hp>
550 goto590                         <ee>
560 pokea+len(a$)+1,90              <bm>
570 pokea+len(a$)+2,105:goto590     <gj>
580 pokea+len(a$)+3,105             <mc>
590 next                            <cc>
600 next                            <dg>
610 fori=399to80step-1              <im>
620 a=peek(b+i)                     <gg>
630 ifa=105thena=233:goto660        <ck>
640 ifa=233thena=105:goto660        <ol>
650 ifa=124thena=123                <nj>
660 pokeb+399-i+480,a               <en>
670 next                            <md>
680 fori=1to4                       <dn>
690 gosub840                        <cm>
700 next                            <pp>
710 pokeze,21:pokesp,29:syscu       <gg>
720 print"oben"                     <km>
730 printtab(29)"hinten"            <pn>
740 printtab(29)"unten"             <fc>
750 printtab(29)"vorne";            <je>
760 fori=b+876tob+996step40         <ml>
770 fori0=0to3                      <nh>
780 pokei+i0,250                    <kf>
790 nexti0,i                        <ph>
800 return                          <ne>
810 ***********************         <ej>
820 *** faerben wuerfel ***         <ac>
830 ***********************         <lb>
840 a=f+72+10*i                     <ai>
850 fori0=0to1                      <ln>
860 a=a+39*i0                       <nm>
870 fori1=0to5                      <nd>
880 pokea+i1,w(i,1)                 <ca>
890 nexti1,i0                       <ba>
900 pokef+875+i,w(i,1)              <fc>
910 a=f+151+10*i                    <gp>
920 fori0=0to4                      <cj>
930 a=a+40                          <af>
940 fori1=0to4                      <bp>
950 pokea+i1,w(i,2)                 <pk>
960 nexti1,i0                       <la>
970 pokef+995+i,w(i,2)              <fp>
980 a=f+79+10*i                     <lk>
990 fori0=0to1                      <ad>
1000 a=a+39                         <hk>
1010 fori1=0to5                     <ab>
1020 pokea+40*i1,w(i,3)             <ig>
1030 nexti1,i0                      <fk>
1040 a=f+434+10*i                   <dm>
1050 fori0=0to4                     <gf>
1060 a=a+40                         <kg>
1070 fori1=0to4                     <fp>
1080 pokea+i1,w(i,4)                <jg>
1090 nexti1,i0                      <ep>
1100 pokef+915+i,w(i,4)             <de>
1110 a=f+433+10*i                   <ia>
1120 fori0=0to1                     <ef>
1130 a=a+39                         <bk>
1140 fori1=0to5                     <el>
1150 pokea+40*i1,w(i,5)             <aj>
```

```
1160 nexti1,i0                                  <ae>
1170 a=f+674+10*i                               <oc>
1180 fori0=0to1                                 <kd>
1190 a=a+39                                     <ek>
1200 fori1=0to5                                 <kb>
1210 pokea+i1,w(i,6)                            <nc>
1220 nexti1,i0                                  <ol>
1230 pokef+955+i,w(i,6)                         <mc>
1240 return                                     <eh>
1250 *************************                  <od>
1260 *** ordnen der wuerfel ***                 <ip>
1270 *************************                  <cl>
1280 w=1                                        <bj>
1290 a=b+393+w*10                               <el>
1300 pokea,30                                   <on>
1310 pokea-b+f,242                              <gb>
1320 ifjoy(1)<128then1370                       <dl>
1330 pokea,32                                   <kd>
1340 w=w+1:ifw=5thenw=1                         <cj>
1350 fori=1to300:next                           <hm>
1360 goto1290                                   <ch>
1370 geta$:ifa$=chr$(13)then1750                <dj>
1380 a0=joy(1)                                  <ii>
1390 if(a0=0)or(a0>8)then1320                   <ah>
1400 if(a0=4)or(a0=6)then1320                   <ge>
1410 fori=1to200:next                           <fh>
1420 ifjoy(1)<>a0then1320                       <mk>
1430 ona0gosub1790,1900,2010                    <oj>
1440 ona0-4gosub2120,2120,2230,234
0                                               <pj>
1450 ad(w)=ad(w)+1                              <pk>
1460 ad=ad+1                                    <ha>
1470 printchr$(19)tab(5)ad(1),                  <jd>
1480 printspc(5)ad(2),                          <np>
1490 printspc(5)ad(3),                          <ol>
1500 printspc(5)ad(4),                          <ph>
1510 printad                                    <ig>
1520 i=w:a0=a                                   <ie>
1530 gosub840                                   <nn>
1540 ifa$="n"thengosub2700                      <ih>
1550 a=a0:a0=0                                  <jk>
1560 fori=1to6                                  <bn>
1570 if(i=3)or(i=5)then1730                     <no>
1580 a1=0                                       <if>
1590 fori0=1to4                                 <fi>
1600 a1=a1+w(i0,i)                              <nc>
1610 next                                       <cc>
1620 i0=40                                      <li>
1630 ifi=4theni0=80                             <cd>
1640 ifi=6theni0=120                            <km>
1650 ifi=2theni0=160                            <em>
1660 fori2=f+829+i0toi2+6                       <jd>
1670 pokei2,peek(i2)and127                      <cm>
1680 next                                       <kp>
1690 ifa1<>344thena0=1:goto1730                 <ed>
1700 fori2=f+829+i0toi2+6                       <ec>
1710 pokei2,peek(i2)or128                       <el>
1720 next                                       <pp>
1730 next                                       <bd>
1740 ifa0=1then1320                             <jj>
1750 return                                     <eh>
1760 *************************                  <od>
1770 *** kippen nach hinten ***                 <gg>
1780 *************************                  <cl>
1790 a0=w(w,4)                                  <di>
1800 w(w,4)=w(w,1)                              <hd>
1810 a1=w(w,6)                                  <ho>
1820 w(w,6)=a0                                  <oc>
1830 a0=w(w,2)                                  <ho>
1840 w(w,2)=a1                                  <bl>
1850 w(w,1)=a0                                  <no>
1860 return                                     <ce>
1870 *************************                  <dd>
1880 *** kippen nach rechts ***                 <ii>
1890 *************************                  <fh>
1900 a0=w(w,3)                                  <ap>
1910 w(w,3)=w(w,1)                              <mc>
1920 a1=w(w,6)                                  <kd>
1930 w(w,6)=a0                                  <ip>
1940 a0=w(w,5)                                  <ai>
1950 w(w,5)=a1                                  <oi>
1960 w(w,1)=a0                                  <pl>
1970 return                                     <ab>
1980 *************************                  <mg>
1990 *** drehen nach rechts ***                 <fb>
2000 *************************                  <hl>
2010 a0=w(w,3)                                  <oi>
2020 w(w,3)=w(w,2)                              <hb>
2030 a1=w(w,4)                                  <nd>
2040 w(w,4)=a0                                  <il>
2050 a0=w(w,5)                                  <bn>
2060 w(w,5)=a1                                  <hd>
2070 w(w,2)=a0                                  <mi>
2080 return                                     <no>
2090 *************************                  <me>
2100 *** kippen nach vorne ***                  <gg>
2110 *************************                  <hn>
2120 a0=w(w,2)                                  <mm>
2130 w(w,2)=w(w,1)                              <bo>
2140 a1=w(w,6)                                  <hm>
2150 w(w,6)=a0                                  <kg>
2160 a0=w(w,4)                                  <ea>
2170 w(w,4)=a1                                  <io>
2180 w(w,1)=a0                                  <ce>
2190 return                                     <ll>
2200 *************************                  <ik>
2210 *** drehen nach links ***                  <lk>
2220 *************************                  <mj>
2230 a0=w(w,5)                                  <jl>
2240 w(w,5)=w(w,2)                              <dk>
2250 a1=w(w,4)                                  <fg>
2260 w(w,4)=a0                                  <hf>
2270 a0=w(w,3)                                  <pc>
2280 w(w,3)=a1                                  <kc>
2290 w(w,2)=a0                                  <jn>
2300 return                                     <jg>
```

```
2310 **************************          <lb>
2320 *** kippen nach links ***           <jk>
2330 **************************          <ma>
2340 a0=w(w,5)                           <ie>
2350 w(w,5)=w(w,1)                       <of>
2360 a1=w(w,6)                           <cn>
2370 w(w,6)=a0                           <pc>
2380 a0=w(w,3)                           <ja>
2390 w(w,3)=a1                           <ff>
2400 w(w,1)=a0                           <da>
2410 return                              <he>
2420 *****************                   <fl>
2430 *** spielende ***                   <cb>
2440 *****************                   <ob>
2450 pokeze,21:pokesp,0:syscu            <al>
2460 ifa$=chr$(13)then2480               <io>
2470 print"geschafft !!!"                <pf>
2480 print                               <am>
2490 print"ende ? (j/n)"                 <dh>
2500 getkeya$                            <gk>
2510 ifa$="j"then2650                    <ci>
2520 ifa$<>"n"then2500                   <np>
2530 scnclr                              <lp>
2540 fori=1to4                           <of>
2550 ad(i)=0                             <jp>
2560 next                                <jg>
2570 ad=0                                <on>
2580 print"standardspiel ? (j/n)         <mk>
2590 poke239,0                           <bg>
2600 getkeya$                            <lc>
2610 ifa$="n"then2650                    <gl>
2620 ifa$<>"j"then2600                   <ci>
2630 gosub3190                           <nd>
2640 a$="a"                              <ld>
2650 scnclr                              <la>
2660 return                              <gk>
2670 **********************              <ab>
2680 *** zufallswuerfel ***              <jb>
2690 **********************              <hl>
2700 fori0=1to4                          <kh>
2710 fori=1to6                           <lm>
2720 a=int(rnd(a)*4)+1                   <le>
2730 ifa=1thena=51                       <ic>
2740 ifa=2thena=70                       <df>
2750 ifa=3thena=103                      <pf>
2760 ifa=4thena=120                      <kp>
2770 w(i0,i)=a                           <mm>
2780 if(i=3)or(i=5)then2920              <ja>
2790 ifi0>1then2870                      <ac>
2800 ifi=1then2920                       <ci>
2810 fori1=1toi-1                        <io>
2820 if(i1=3)or(i1=5)then2840            <bo>
2830 ifa=w(1,i1)theni1=99                <og>
2840 next                                <mi>
2850 ifi1=100then2720                    <dk>
2860 goto2920                            <ia>
2870 a0=0                                <hc>
2880 fori1=1toi0-1                       <bn>
2890 ifa=w(i1,i)thena0=1                 <ac>
2900 next                                <ea>
2910 ifa0=1then2720                      <jd>
2920 nexti,i0                            <dk>
2930 forw=1to4                           <kc>
2940 fori=1to6                           <hg>
2950 a=int(rnd(1)*4)or1                  <mg>
2960 fori0=1toa                          <dk>
2970 onigosub1790,1900,2010,2120,2
230,2340                                 <do>
2980 nexti0,i,w                          <lk>
2990 return                              <ab>
3000 ********************                <ij>
3010 *** anfangswerte ***                <ao>
3020 ********************                <mi>
3030 color4,1     :rem randfarbe         <nb>
3040 color0,1     :rem hintergrund       <ad>
3050 color1,16,5:rem zeichenfarbe        <ea>
3060 scnclr       :rem loeschen bild     <ka>
3070 b=3072       :rem bildram           <dd>
3080 f=2048       :rem farbram           <pk>
3090 ze=205       :rem cursorzeile       <fe>
3100 sp=202       :rem cursorspalte      <fe>
3110 cu=55464     :rem cursor setzen     <jb>
3120 dimw(4,6)    :rem wuerfelfarben     <kp>
3130 gosub2580                           <jg>
3140 ifa$="n"thengosub2700               <po>
3150 return                              <ec>
3160 *****************************       <oe>
3170 *** setzen standardfarben ***       <hl>
3180 *****************************       <pp>
3190 restore                             <jb>
3200 fori=1to4                           <lh>
3210 fori0=1to6                          <kj>
3220 reada                               <nc>
3230 w(i,i0)=a                           <ph>
3240 nexti0,i                            <gb>
3250 data51,51,120,70,103,51             <bf>
3260 data120,120,51,103,70,70            <gd>
3270 data70,120,51,51,103,120            <io>
3280 data103,120,70,103,70,51            <pf>
3290 return                              <fl>
3300 end                                 <gb>
3310 rem ende ende ende                  <dk>
```

Jetzt perfekt: Unser Checksummer

Hatte bisher unser Checksummer an Buchstabenvertauschungen nichts auszusetzen, so zeigt er sich nun nicht mehr so kulant.

Ob Sie mit der alten Version nun eingegeben hatten:

10 print"ab"
oder
10 print "ba",

der Checksummer brachte in beiden Fällen die Prüfsumme < gk >. Leicht kann es vorkommen, daß beim schnellen Tippen, besonders im Zehnfingersystem, die Taste, die eigentlich erst als übernächste drankommen sollte, ein wenig zu früh erwischt wird. Dem Checksummer, der lediglich die Ascii-Werte der Buchstaben addierte, konnte dieses natürlich nicht auffallen. Was also tun? Ob etwas früher oder später addiert wird, ändert nichts am Resultat der Summe. Anders ist es, wenn man zwei Verknüpfungsarten kombiniert. So ist z.B. 2*30+40 etwas anderes als 2*40+30. Und genau dieses war dann die Lösung. Die Summe wird nun einfach durch eine Linksverschiebung vor jeder Addition verdoppelt. Dadurch, daß im Falle, wenn das Ergebnis größer als 255 ist, der dabei entstehende Übertrag als Wert 1 zusätzlich addiert wird, verflüchtigen die Werte der am Anfang der Zeile gefundenen Codes sich nicht nach 8 weiteren Zeichen. Damit bleibt nicht nur die Aussagekraft der Prüfsumme voll erhalten, sondern erfährt sogar eine erhebliche Steigerung. Und vor allen Dingen wird nur eine klitzekleine Änderung erforderlich, die dieses zu vollbringen, in der Lage ist. Ein einziges Byte ist nur zu ändern. Wir tun dieses mit "poke 345,10" in der Zeile 470. Dadurch wird das hier ursprünglich ansässige CLC (Clear Carry) durch ASL (Arithmetik Shift Left) ersetzt. Die nachfolgende Addition mit ADC (Addiere mit Carry) addiert den Ascii-Code des gefundenen Zeichens und den nach links herausgeshifteten Übertrag. Da einige unserer Leser beklagten, daß das Checksummerlisting nachher noch im Programmspeicher stehen würde, haben wir diesem noch mit einem "new" abgeholfen. New bzw. neu ist nun folgendes.

10 print"ab" ergibt die Prüfsumme<jd>
10 print"ba" die Prüfsumme<jf>

Sie brauchen den Checksummer nicht neu einzutippen. Alles, was Sie tun müssen, ist, die Zeile 470 anzufügen. An der Bedienung des Checksummers hat sich nichts geändert. Die Eingabehinweise bleiben daher wie gehabt.

EINGABEHINWEISE

Am rechten Rand jedes Listings, jeweils am Ende einer Eingabezeile, finden Sie zwei Buchstaben zwischen einem Kleiner- und einem Größerzeichen eingeschlossen. Diese dürfen Sie nicht mit in Ihr Listing eintippen, sondern sie dienen Ihnen zur Überprüfung Ihrer Eingabe.

Zwischen dem Kleiner- und dem Größerzeichen am rechten Rand befinden sich zwei Buchstaben. Mit einem speziellen Programm können Sie beim Eintippen Ihre Eingabe auf ihre Richtigkeit überprüfen. Dieses Programm, der Checksummer, sorgt nämlich dafür, daß nach erfolgter Zeileneingabe am linken oberen Bildschirmeck zwei Buchstaben ausgegeben werden. Wenn diese Buchstaben nicht mit den vorher erwähnten Buchstaben in unserem Listing übereinstimmen, so können Sie davon ausgehen, daß Sie sich vertippt haben und können sich so die Zeile nochmals näher ansehen, ob Sie Ihren Eingabefehler finden. Wenn Sie dann alles richtig getippt haben, so stimmen die Buchstaben überein und Sie können sich getrost der nächsten Zeile zuwenden.

ERST SICHERN, DANN AUSPROBIEREN

Das Checksummerlisting hat noch keine Prüfsummen. Seien Sie deshalb besonders aufmerksam, daß alles paßt und speichern Sie dieses Programm unbedingt ab, bevor Sie es starten! Bei einem Tippfehler würde es sich wahrscheinlich auf Nimmerwiedersehen verabschieden und Sie müßten die ganze Arbeit vermutlich nochmals machen. Wenn Sie es gestartet haben, so geschieht nichts Besonderes. Der Computer meldet sich einfach kurz darauf mit „READY", und das war auch schon alles. Alles sollte nun wie immer funktionieren, mit der kleinen Ausnahme, daß nunmehr nach jeder Eingabe im Direktmodus eine Prüfsumme erscheint. Nehmen Sie zum Testen irgendeine kurze Basiczeile aus unserem Heft her und testen sie aus. Wenn die Summen übereinstimmen, so können Sie sich freuen, denn Fehler beim Abtippen werden Ihnen nun in Zukunft viel weniger passieren, als vorher.

EINER FÜR ALLE, EIN ECHTES UNIVERSAL-PROGRAMM

Unseren Checksummer können Sie verwenden, ob Sie einen C16/116/Plus4 oder ob Sie einen C64 oder gar einen C128 haben. Nur müssen Sie beim letzteren beachten, ob Sie auch wirklich im 40-Zeichenmodus sind. Nachdem Sie den Checksummer geladen und gestartet haben, können Sie Ihr Basicprogramm eingeben wie gewohnt, Sie können es abspeichern, Sie können auch laden, Sie können Kürzel verwenden und, ob Sie ein paar Leerzeichen mehr oder weniger verwenden, der Checksummer läßt sich dadurch nicht aus der Fassung bringen. Ein bißchen Vorsicht sollte man allerdings walten lassen, wenn man Programme eingetippt hat, in denen Peeks und Pokes vorkommen. Es wird zwar nicht besonders häufig vorkommen, aber es könnte bisweilen ge-

```
10 rem =checksummer==c16 c64 c128==
20 rem (p) 05/87 commodore welt  ==
30 rem =============================
40 rem (c) alfons mittelmeyer     ==
50 rem                            ==
60 rem c16/116/plus4              ==
70 rem c64                        ==
80 rem c128 (40-zeichen)          ==
90 rem =============================
100 rem ---------------------------
110 rem   grundroutine (c16)
120 rem ---------------------------
130 data165,059,072,165,060,072,032
140 data086,137,104,133,060,104,133
150 data059,152,072,160,000,165,020
160 data024,101,021,170,024,144,011
170 data201,032,240,006,138,024,113
180 data059,234,170,200,177,059,234
190 data208,240,169,031,072,138,074
200 data074,074,074,072,138,041,015
205 data072,169,031,072,162,003,104
210 data024,105,129,157,000,012,202
220 data016,246,104,168,096
230 lt=peek(772):ht=peek(773)
240 fori=312to386:readx:pokei,x:nex
t
250 iflt<>124then350
260 rem ---------------------------
270 rem        anpassung c64
280 rem ---------------------------
290 fori=312to317:pokei,234:next
300 fori=321to326:pokei,234:next
310 fori=1to6:readad:readx:pokead,x
:next
320 poke380,4:poke319,lt:poke320,ht
:goto430
330 data346,121,347,000,348,002
340 data351,185,352,000,353,002
350 iflt<>13then430
360 rem ---------------------------
370 rem anpassung c128 (40 zeichen)
380 rem ---------------------------
390 restore410:poke332,22
400 poke335,23:goto310
410 data313,061,316,062,323,062
420 data326,061,347,061,352,061
430 poke772,056:poke 773,1
440 rem ---------------------------
450 rem ergaenzung 10/87
460 rem ---------------------------
470 poke 345,10:new
480 rem ===========================
490 rem = fuer hefte cw 7/87 bis   =
500 rem = cw 9/87 sowie cw128 5/87=
510 rem = und c16 6/87 ist die     =
520 rem = poke-anweisung in zeile =
530 rem = 470 wegzulassen          =
540 rem ===========================
```

schehen, daß nach dem Laufenlassen eines Programmes weder der Checksummer noch sonst etwas mehr funktioniert, auch wenn dies bisher ohne Checksummer nicht der Fall gewesen sein sollte. Also bitte sichern Sie in jedem Falle Ihre Programme, bevor Sie sie ausprobieren.

Ein paar Dinge sollten Sie noch wissen. Wir drucken in unseren Listings des öfteren Punkte statt Leerzeichen. Wenn Ihnen nun aber Leerzeichen besser gefallen, so liefert der Checksummer natürlich eine falsche Summe. Wenn Sie diese Richtigkeit überprüfen wollen, so können Sie dies tun, indem Sie sie zuerst einmal so wie im Heft abtippen, und nachher, nachdem Sie sie nachgeprüft haben, einfach wieder die Punkte durch Leerzeichen ersetzen.

A. Mittelmeyer

MONITOR

CHECKMON

```
40 rem checkmon ===============c16  <cn>
50 rem (p) commodore welt team      <ke>
60 rem ===========================  <nk>
70 rem (c) by  a.mittelmeyer        <ag>
80 rem c16/116/plus4                <ki>
90 rem ===========================  <jg>
100 fori=312to398:reada             <ei>
110 pokei,a:next                    <ep>
120 data 132,218,108,219,0,132,219  <oe>
130 data 164,218,76,75,236,201,62   <nk>
140 data 208,249,165,161,10,101     <jc>
150 data 162,160,7,10,113,161,136   <ej>
160 data 16,250,133,216,169,30,133  <oh>
170 data 217,169,62,160,97,208,220  <mk>
180 data 198,217,208,218,160,105    <ai>
190 data 208,212,201,13,240,4,164   <ha>
200 data 218,24,96,169,60,160,68    <lh>
210 data 32,61,1,165,216,32,16,251  <ec>
220 data 169,62,160,5,208,2,169,32  <om>
230 data 32,75,236,136,208,248,169  <ol>
240 data 13,208,176,219,68,220,1    <go>
250 data 804,56,805,1               <hn>
260 fori=1to4:reada:readb:pokea,b   <lm>
270 next:new                        <ji>
280 rem ========e=n=d=e==========   <cc>
```

"CHECKMON" ist eine unerlaessliche Hilfe zur Eingabe von Maschinenprogrammen. Laden und starten Sie "Checkmon" und gehen dann mittels MONITOR in denselben. Wenn Sie sich nun z.B. mit 'M1000' einen Speicherbereich ansehen, oder Hexzahlen eingeben, so erscheint rechts die Pruefsumme anstatt der Asciicodes.

C16/P4: NOVALOAD-COPY (Tape to Tape). Autostart und Turbo werden absorbiert. ACHTUNG! Das Progr. darf nur f. Sicherheitskopien genutzt werden. Zusätzlich für Sie NOVA-TURBO 30,– DM/Bar. Müller-electronic Schuhstr. 5, 3110 Uelzen 1

VERKAUFE aus allen Sparten Software (Anfrage). Jede Disk (f. C16/+4) 10,– DM. Matthias Hagen, Neustr. 40, 4018 Langenfeld

---

VERKAUFE Spiele u. Anwenderprogr. f. C16/+4, jede Diskette 10,– DM u. orig. Markt&Technik-Progr.-Sammlungen I + II + III u. High-Screen-CAD f. 20,– bzw. 25,– DM incl. Anleitungen. Tel. 0231/270660

VERKAUFE Prg.-Disk aus CW-SA 4/87, 6/87; C. m. SA 3/87, 4/87; 64er SA 14/87, Textword II Vers. 2,0 m. Handb. u. Brosch.; Tips&Tricks, ADM, TW II 30,– DM; orig. ACE 10,– DM (frei Haus). Tel. 0591/2086

SUCHE Progr. f. +4: Superbase, Mikrocalc, Script Plus, Calc Plus u. a. Anwenderprg. (Tausch od. Kauf). Michael Gode, Ellernstr. 21, 3032 Fallingbostel 1

FREEWARE auf Kass. (ca. 40 Prg.) f. C16/116/+4 mit u. ohne 64 K. Liste gg. 10,– DM (System angeben). Ulrich Watzlawik, Sudetenstr. 32, 8882 Lauingen

---

VERKAUFE Mathe-u. Graphik-Master. 8,– DM, Kass. u. Rückumschlag an Marc Kluetig, Schwarzenberger Str. 13, 5270 Gummersbach 1, Tel. 02261/66841

VERKAUFE +4 m. Floppy 1551, Datasette, Joystick u. Bücher an Meistbietenden. Rudolf Dercks, Fürstenbergstr. 8, 4150 Krefeld 1

VERKAUFE C16-Software auf Kass. od. Disk. Spiele ab 1,– DM. Liste gg. 2,– DM auf Tape od. Disk bei Gerhard Winter, Am Schwänzchen 23, 5300 Bonn 1

VERKAUFE C16-Progr.-Sammlung mit 72 Spielen (z.B. Fire Galaxy, Monty 1 u. 2 etc.) für 35,– DM auf Disk. Info gg. Rückporto bei Kai Uwe Boß, Ilmenweg 5, 6422 Herbstein, Tel. 06643/8286 ab 18 h.

SUCHE +4-User, der mit mir Programme tauscht (Spiele u. Anwenderprogramme). Liste an Norberg Albrecht, Grundweg 1, 2000 Norderstedt

---

C16/P4-USER! Orig.-Software-Rest. Liste gg. Freiumschlag. Nur Orig.-Kass., aber spottbillig. Harald Scheel, Kollaustr. 178, 2000 Hamburg 61

C16/116/P4! HEX-Tastatur f. die komfortable Eingabe v. MC-Programmen m. TEDMON od. Checksummer. In BASIC als 2x16 Funktionstasten verwendbar. DM 60,–. Hans-D. Veit, Rennweg 21, 8441 Aiterhofen

---

SUPER-ANGEBOT: STAR GAMES 60 zum C16/116/Plus4 für 39,– Sfr od. 39,– DM. Auf 4 Kass. finden Sie 60 interessante Spiele. Lieferung erfolgt per Nachn. Kostenlose Info bei: M. Greifenhagen, Stöckelstr. 8, CH-8610 Uster

---

*** SYSTEMWECHSEL *** Plus 4 f. 100,– DM; Floppy 1551 f. 100,–; Drucker MPS 801 f. 250,– DM; Monitor 150,– DM; 2 Joyst. 20,– DM; 3mal M+T-Disk u. Sonderhefte 75,– DM; Lightpen 40,– DM; Script+ 40,– DM; ELCAD u. HiRes-HC 65,– DM; 16C-Welt 25,– DM. K. Gaisser, 07191/44211

COMMODORE USER CLAN IN DER DDR sucht für seine C16/116/Plus4-Freaks Computerzeitschriften, Bücher, Kopien sowie Hardware aller Art u. Kassettenschrott, da keine Verwandten in der BRD. Wir bedanken uns recht herzlich. Ludwig Steuner, Teutoniaweg 1, DDR 9272 Gersdorf

---

SUCHE alte Computer-Zeitschriften (CW, Happy Computer, 64er usw.). Benötige unbedingt Happy Computer Nr. 1/86! SEHR WICHTIG! Angebote an: Throsten Boese, Pappelweg 23, 2300 Kiel 1

SUCHE VC 1520 Plotter/Printer. Wo bekomme ich die 4-1/2-Zoll-Additionsrollen? Paul Brune, Schreinerstr. 4, Tel. 05242/34689

TAUSCHE Software f. Plus4. Biete ca. 350 Games u. 100 Anw. Jede Zuschrift wird beantwortet. Schickt Eure Listen an Thomas Weisheit, Pommrück 6, 6431 Hauneck

---

VERKAUFE Liste m. 35 Pokes u. 50 Startadressen v. bekannten Spielen f. C16/116/P4 f. nur 10,– DM-Schein an: Jens Borau, Ahornstr. 4, 3549 Volkmarsen

---

C16/P4: VERKAUFE 6 Orig. Spiele (Tape) f. 50,– DM. 2 Spiele nur f. 64 KB. Tel. 05674/4645

Wer hat DFÜ-Erfahrungen m. Plus4? Bitte melden bei Rainer Bielefeld, Neustadtring 23, 3300 Braunschweig, Tel. 0531/503589

PLUS4/C16. Orig.-Spiele (Disk) zu verk.: Profipack II 64K 80,– DM; Music Master, Grafikdesigner, Micro Text 2.0 je 10,– DM; Plus4 Sextett 20,– DM; alles zzgl. Porto. Heinz Weissmann, Im Leitle 2, 8570 Pegnitz, Tel. 09241/2627

VERKAUFE 64er-SH 14 (C16 etc.) m. Prog.-Disk f. 20,– DM; CW-SH 4/87 f. 6,50 DM; Comp. mit SH 1/87 f. 6,50 DM; Comp. mit SH 1/87-3/87 a 3,80 DM; Comp. mit 12/86-3/87 a 1,70 DM; 64er 12/86, 2/87, 3/87, 6/87 a 3,80 DM. Kaiser, Gorch-Fock-Str. 7, 2057 Reinbek, Tel. 040/7104765

Gehäuse f. Plus4, C16 f. DM 20,–. Div. Software auf Eprom. Viele Ersatzteile f. C16 u. Plus4. Info gg. Freiumschlag. Harald Hobbelmann, Junkernkamp 18, 2822 Schwanewede, Tel. 04209/5390

SUCHE Floppy 1541 od. 1551 u. Farbmonitor od. Farbfernseher. Nur funktionstüchtig. Zahle je bis zu 150,– DM. Tel. 02261/48940 ab 17 Uhr

---

HAST DU PROBLEME m. dem C16/Plus4? Du bist Anfänger? Wir helfen. Auch Fortgeschrittenen. Wir bieten Clubzeitung, Infos, jede Menge Tips u. Tricks. Am besten heute noch Info gg. 1,60 DM bei CIG, c/o K.-D. Schindler, Luciusstr. 10, 623 Frankfurt 80, anfordern.

PLUS4, Floppy 1551, Datasette, Monitor, Joyst., div. Bücher u. Zeitschr., Anwenderpr. z.B. Micro Text, Datei, Kalk usw., Spiele z.B. ACE, Bongo, Demol., Strip Poker usw. wg. Systemwechsel zu verk. VB 400,– DM. Lars Winter, Schloßgebiet 4, 2320 Plön, Tel. 04522/3753, 17-18 Uhr

---

Wer kann mir helfen? Suche komplette Liste, um selber Programme u. Spiele zu schreiben! Bin quasi noch Anfänger! Unkosten werden übernommen! Interesse für C16 m. 64K u. C64. Wer kann mir helfen? Hans Jürgens, Provinzstr. 111, 1000 Berlin 51

AUFGEPASST! Verkaufe Plus4, Datasette, 1 Joy, 11 Comp.-Hefte, 22 Spiele, 3 Handbücher, 6 Mon. alt, Preis nach VB od. Tausch gg. C64. Tel. 0201/466338

C16/116/Plus4. Entwicklung v. Rechenprogrammen jeder Art. Statistik u. Kalkulation! Martin G. Maaß, Kulbrockstr. 2, 4800 Bielefeld 14

VERKAUFE f. C16/Plus4 Spiele u. Anwenderprog., 20 Prog. zu 10,– DM. VERKAUFE C16-Orig.-Spiele zum halben Preis z.B. P.O.D, R16, Attack. Schreibt an: C. Schnoor, Lausanner Str. 129, 2800 Bremen 44

VERKAUFE: C116/C16/Plus4. Anschlußkarte f. Tastatur m. 10er Block (CBM 600/700), Erw. Karte m. 2 Steckplätzen f. Expans.-Port, z.B. f. Eproms. Brenne Ihre Eproms. Suche Hard- u. Software aller Art. Tel. 040/7125694

Plus4/C16/C116: Biete 50 Spiele, div. Anwend.-Prog. f. DM 20,– (Disk). Schein od Scheck. Suche Kopierprog. v Kass. auf Disk. Listings gg 80 Pf. in Briefmarken.

SOFTWARE 1988! Spitzensoftware f. den Commodore 14/+4 bietet Ihnen dieser Software-Katalog 88 zu absoluten Niedrig-Preisen! Gratis-Info, Katalog nur 1,30 DM. P. Schäfers, Riekestr. 5, 4402 Greven 1

ACHTUNG! Günstig! Verkaufe C16 m. 64 KB, 1551, 1531, Quickshot II, Ada., 60 Disk. (voll), 7 Originalkassetten u. Basickurs, Zeitschriften! VB 600 DM. Oliver Berger, Lindenstr. 7, 8260 Mühldorf

---

AN ALLE PLUS/4-Besitzer! Tausche Commodore 16/64 Kb u. Sinclair ZX 81 gegen einen funktionstüchtigen PLUS/4; 2 Computer für einen! Beide Geräte vollkommen in Ordnung und m. Handbüchern. Tel. 05475432

SCHÜLERIN ohne Einkommen (Taschengeld 20,– DM/Mon.) hat Zuzahlungszusage a 100,– DM u. sucht intakte Floppy u. Drucker A4 für +4. Angebote: Melanie Schimmelpfennig, Kleeweg 23, 4830 Gütersloh, Tel. 05241/46235

SUCHE HAPPY Computer Sonderheft 5 (Programmiersprachen) sowie 64er Sonderheft 12 (PRG-Sprachen) u. 7/85 (DFÜ + Anwendungen). Michael Gunske, Westernheide 16, 2811 Martfeld

---

ARMER C-Einsteiger aus der DDR sucht +4 od. C16 (64 KB) u. Datasette f. max. 100,– DM. Wer gibt kostenlos defekte Joyst. od. anderen „C-Schrott“ ab? Bettina Nagy, Lange Gasse 51, 7988 Wangen, Tel. 07522/1882

ORIG. Programme (Kass.) f. C16 u. 64K. Winter-, -Sommer-Olympiade, Quiwi, Musikmaster, Microkalk, Grandmaster, Turbo-Tape, Progr.-Kass. Compute Mit, Joyad. Kabel f. 210,– DM, Joystick 20,– DM. Dieter Rapp, Stahlackerweg 36, 7311 Ohmden, Tel. 07023/5775 ab 19 h.

SUCHE f. C16/+4 Freeware u. Public-Domain-Software (auch abgetippte Listings). Bes. ges.: Programmiersprachen, Anwendungen,Adv. u. Strategiespiele. Michael Gunske, Westernheide 16, 2811 Martfeid

---

AN ALLE Amiga- u. PLUS-4-User im Raum Bad Kreuznach: Wir suchen für unseren Club noch Mitglieder. Info bei Holger Scherer, Jungstr. 25, 6550 Bad Kreuznach

---

SUCHE f. PLUS4/C16 Bücher (zahle halben NP) u. Anleitungen f. Jump Jet, Frank B.B., Spy vs Spy u. TED) u. andere Chips f. PLUS4/C16. Carsten Schnoor, Lausanner Str. 121, 2800 Bremen 44, Tel. 0421/422460

SUCHE Tauschpartner aus der ganzen Welt für PLUS4/C16/116; habe üb. 1300 Progr. Carsten Schnoor, Lausanner Str. 121, 2800 Bremen 44, Tel. 0421/422460

COMMODORE +4 incl. Laufwerk 1551 u. 10 Disketten m. Spielen u. Programme zu verkaufen, VB 380,– DM. Tel. 0761/29970

VERKAUFE Austro Speed Compiler, 1-2-3 CAD u. viele neue Games f. +4. R. Vukecivic, Lollmannshof 34a, 4800 Bielefeld

---

VERKAUFE C16 orig. Spiele 50% billiger als NP. Liste anfordern bei Alexander Käss, Hummelstr. 7, 7987 Weingarten

---

SUCHE Tauschpartner f. +4 auf Disk od. Kass. Suche Farbmonitor f. +4. Hans-Jürgen Kinateder, Salinenstr. 6, 6927 Bad Rappenau, Tel. 07264/6957

SUCHE Kontakte zu +4-Usern; Spez. Interessen: Anwenderprogramme, Hardware, DFÜ etc. Wer hat Erfahrung mit der VC 1581 am +4? Thomas Rudolph, Eisenberger Str. 9, DDR-8023 Dresden

---

VERKAUFE Lottoprog. m. hoher Wahrscheinlichkeit f. Kass. od. Disk f. 10,– DM. Suche Tauschpartner f. C16/+4 (nur Disk). Suche Backupprogr. f. 2 Floppys. Dirk Freund, Rosenstr. 7, 3563 Elmshausen, Tel. 06466/7942

VERKAUFE orig. Kass. u. Disk. von Kingsoft, Markt & Technik u.a. zum halben Neupreis (Spiele, Anwenderprogr. etc.). Theo Steinberg, Mehlerweg 5, 4600 Dortmund 13, Tel. 0231/270660

VERKAUFE Winter-Games u. „One man and his droid“ auf Kass. f. 250,– ÖS. Suche kaputte Joysticks. Harald Pusar, Sonnengasse 28, A-9020 Klagenfurt

---

VERKAUFE orig. Spiele auf Tape f. C16/ +4; ACE 20,– DM; C. Slalom 20,– DM; F.B. Boxing 15,– DM; R.F. Sky 15,– DM; Booty 7,– DM; Torpedo Alley 5,– DM. Bianca Hauschild, Fährdeich 109, 2101 Hamburg 96

BIN EINSTEIGER und suche Kontakt zu C16/ +4-Usern zwecks Infotausch und gegenseitiger Hilfe. Tel. 0781/42523

LOGO f. +4, C16 (64 K), Modul, Disk u. Handb. f. 50,– DM bei Vorkasse (Nachn. 5,– DM). Div. Software auf Steckmodul. Info gg. Freiumschlag. Harald Hobbelmann, Junkerkamp 18, 2822 Schwanewede, Tel. 04209/5390 ab 18 h.

ERSATZTEILE f. +4/C16. Evt. Reparaturhilfe. Div. Data-Becker-Bücher zu verkaufen. Suche 80-Zeichen-Monitor. Harald Hobbelmann, Junkernkamp 18, 2822 Schwanewede, Tel. 04209/5390 ab 18 h.

VERKAUFE 15 C16-Orig.-Spiele, z.B. Bandits at Zero, Ghost Town u.a.; nur zusammen f. 100,– DM. Suche Floppy 1551 bis 150,– DM. Peter Wachter, Mühlenweg 10a, 6900 Heidelberg

---

BESITZEN Sie Turbotape von Kingsoft od. ETP? Wünschen Sie sich hierfür einen Autostart? 10,– DM an H. Lindenzweig, Siegwohrstr. 1, 2913 Apen

---

VERKAUFE Spiele für C16/Plus4 für ganz wenig Geld. Einfach nur Liste anfordern (gg. 80-Pf.-Briefmarke): Michael Lenhardt, Im Birkengrund 77, 6050 Offenbach/M.

PLUS4 zu kaufen gesucht. Angebote an Marion Schwalbe, Tegelburg 11, 2351 Boostedt. Tel. 04393/1586

C16/+4-SUPERPROGRAMME, Spiele, Anwender, Kopierprogr. Gratis-Info m. Tips u. Tricks. Liste anfordern. Th. Görtz, Friedr.-Ebert-Str. 113, 6103 Griesheim

---

hrc e.V. – Plus4 * C16 * C116 * Anwender: Der Verein speziell für uns! Mit Zeitschrift u. Software-Bibliothek. Info beim hrc e.V., Bauerland 15, 4800 Bielefeld 1

SUCHE COMPUTE Mit u. Sonderheft f. C16/Plus4, auch C16/Plus4-SPECIAL-Hefte u. andere Lit. Schickt Eure Listen u. Preise bitte schnell an: Wolfgang Polenda, Brückenstr. 15, 5350 Eu.-Roitzheim

---

C16/PLUS4! Direkt aus USA. Freesoft, ca. 2.500 Block z.B. Druckersteuerung, Biorhythmus, Black Jack, Kalenderdruck. Zusätzlich: Lotto, Vokabelprog., monatl. Kosten, Etikettendruck. Schickt 20,– DM an: Wolfgang Koß, Beringstr. 29, 2215 Hademarschen

---

Computerzeitschriften RUN, 64er, Data Welt, Input 64 jahrgangsweise zu verkaufen. Außerdem Spektrum, Bild der Wissenschaft, Physik in unserer Zeit, mehrere Jahrgänge, preiswert zu verkaufen. Tel. 07231/481493

RELATIVE Dateiverw. der Spitzenklasse f. +4/C16/116. Freie Maskenerstellung. Bis zu 1500 Datensätze (160 kB). Mit Handbuch 19,50 DM + Porto u. Nachnahme. Jürgen Vowerk, Dörener Weg 29a, 4790 Paderborn, Tel. 05251/58423

+4-USER aus der DDR sucht Superbase f. +4 u. Alles über Superbase u. Script/Plus; bin auch an Software u. Erf.-austausch interessiert. Karbe, Th.-Brugsch-Str. 50, DDR-1115 Berlin

---

Kaufe laufend defekte C16 & Plus/4, Floppys und Datasetten. Angebote mit Preisvorstellung und Fehlerangabe. J. Braunroth, Moordorferstr. 30, 3057 Neustadt

---

WER KENNT Execupopt 3000 – 4000 Data Terminal Computer Transceiver und hat Unterlagen dafür, um sie mir zu leihen/fotokopieren? Walter Sewerin, Theodor-Heuss-Ring 52, 5090 Leverkusen 1, Tel. 0214/56217

VERKAUFE Spiele f. +4, z.B. Mission, Ace 2, Zork 2 etc. Info gegen Rückporto bei Sven Kuessner, Bredde 6, 5758 Fröndenberg/Ardey

---

C16/64K, verstärktes Netzteil, Floppy 1541 II (gara.); Joystick, Datas, I/O-Port u. Turbo + Steckpl., Bücher, Zeitschriften, ROM-Listing, Basic-Kurs, Ace, Spitzensoftware, NP 1600,– DM, VB 800,– DM. H. Bosshammer, Heckenweg 2, 4920 Lemgo 1, Tel. 05261/87261

SUCHE Tauschpartner f. +4. Nur Top-Software gesucht. Suche auch Bücher u. Hefte f. +4. Listen an Lars Weißbrodt, Erpener Weg 14, 4503 Dissen

---

VERKAUF/TAUSCH/KAUF – C16/Plus4. Anwender, Games, Freesoft, Info bei Jürgen Cissarek, Giebelstr. 5, 4650 Gelsenkirchen. 100% Antwort.

C16 + 64 K, 1531, Drukker, viele orig. Anwender- u. Spiel-Progr. (64 K, überw. Kingsoft), Joysticks sowie ca. 3 kg Literatur; NP über 800,– DM, VB 300,– DM. Bernd Mackeldanz, Kanzlerstr. 12, 4000 Düsseldorf 30

---

30 GAMES für C16/116/+4 auf Tape für 10,– DM (incl. Porto). Dietmar Neumann, Trierer Str. 398, 5100 Aachen

---

TAUSCHE C16/116/+4 (64K)-Spiele u. Anwender-Progr.; gebe auch Tips. Liste an Andy Grethler, Müllheimerstr. 24a, 7858 Weil/Rhein, Tel. 07621/74946

SUCHE Soft- u. Hardware f. +4/Centronics-Drucker, bis 50,– DM: Floppy bis 150,– DM. Tel. 06221/801863

SUCHE Tauschpartner f. +4-Software. W. Krüger, Volksdorfer Weg 32, 2075 Ammersbeck

---

VIELE GRÜSSE an Thomas in Poppenbüttel. Schalt die Kiste auch mal aus! Wolfgang Krüger, Ammersbek

---

BÜROANWENDUNG f. C16/+4: 6 hilfreiche Anw.-Progr. f. Heim- od. Büroarbeit, 20,– DM (Scheck od. Barzahlung). Guido König, Schlenderhanerstr. 10a, 5000 Köln 60

SUCHE Computerclub f. C16. Infos und/oder Probeheft einer Clubzeitschrift an M. Greifenhagen, Stöcklerstr. 8, CH-8610 Uster

---

SUCHE Drucker f. C16. H. Seitz, St.-Lorenz-Str. 3, 8434 Berching

---

VERKAUFE ZX-Spektrum 128 KB, m. Kass. 100,– DM: suche Progr. Monopoly f. C16/+4. Suche Tauschpartner f. Anwenderprogr. f. C16/+4. Günter Mayer, Hainstr. 12, 7070 Großdeinbach

---

GRÜSSE an die FTA im PA 3 Hamburg. Norbert Albrecht

VERKAUFE CHIP 251641-02 und CHIP 318006-01 f. je 19,90 DM (C16); f. Einsteiger Data Becker f. 19,90 DM. Lieferung nur gg. Nachnahme, Porto u. Versandkosten.
SUCHE Computerclub f. C16. M. Greifenhagen, Stöcklerstr. 8, CH-8160 Uster

---

SUCHE Datasette 1531 oder 1530 m. Cassettenportadapter. M. Greifenhagen, Stöcklerstr. 8, CH-8610 Uster

---

VERKAUFE Keyboard C16 m. G. f. 29,90; Keyboard ohne G. f. 19,90 DM; Verbindungskabel TU-C16/116/+4/C64 f. 9,90 DM. Lieferung nur gg. Nachnahme, Porto u. Versandkosten. M. Greifenhagen, Stöckerlstr. 8, CH-8610 Uster

---

WIR NEHMEN noch Mitglieder auf. Jede Menge Infos, Tips u. Tricks für C16/+4 und Clubzeitschrift. Grafiken für HSC und Zeichensätze für 6,– DM/Diskette. Info gg. 1,60 DM CIG Frankfurt, Ruciusstr. 10a, 6230 Frankfurt/Main 80

VERKAUFE Balkengrafik-Progr. auf Kass. f. 10,– DM/Stck. Geld u. Notiz (ob Farbe od. s/w) an A. Gebhard, Kiefernweg 12, 3113 Suderburg 1

TAUSCHE Software f. +4, Spiele u. Anwenderprogr. Suche Citizen IDP 560. Paul Kasparbauer, Burgweg 1, 8376 Altnussberg

---

VERKAUFE +4 m. Floppy 1551, Datasette, Mouse, Joystick, 2 M&T-Progr. (High-Screen-CAD u. Programms. II), 20 Computerzeitschriften, Software, Zubehör Handbücher f. VB 570,– DM (alles neuwertig). Tel. 04136/8484

Das MPS 803
PRINTKIT V

* Deutscher Zeichensatz
* Unterlängen
* *Kursivschrift*
* Fettschrift
* Unterstreichung
* fünf weitere Sonderzeichen
* höhere Druckgeschwindigkeit

Lieferung umfaßt alle notwendigen Bauteile und ausführliche Einbau- und Bedienungsanleitung.
PRINTKIT V DM 85,- (NN + DM 4,-)
(Mit Einbauservice DM 125,-)
* Speichererw. C16 ab DM 35,-
INFO gegen Porto!

Electronic VersandHandel
B.Döntgen
Lerchenweg 6
4100 Duisburg 14

An uns und/über uns

FRAGEN ZU BASIC-BEFEHLEN

1. Als ich nachsehen wollte, wie ein Programm funktioniert, stieß ich auf eine IF-THEN-Anweisung. An sich kein Problem, aber mitten drin ein AND und ein OR. Man liest zwar viel über AND- und OR-Gatter, aber keiner erklärt, was das ist, wo ich sie anwenden kann, und so weiter. und so weiter.
2. Wozu ist eine DEF FN-Anweisung gut und wie funktioniert sie? Das Handbuch gibt wie immer „hervorragende" Erklärungen.
3. Was nützt es, eine Variable zu dimensionieren? Ich habe das bis heute leider nicht verstanden, vor allem, weil das Wort „dimensionieren" sehr irreführend ist. Am besten kann man es wahrscheinlich mit einem Beispiel erklären, aber bitte nicht so eins wie im 128er-Handbuch!
4. Man sieht in fast jedem Programm, daß eine schon früher benutzte Variable plötzlich hinter einer neuen in Klammern steht. Heißt das, daß jetzt die neue den Wert der alten übernommen hat? Was nützt so etwas?

Fabian Dohmes
Alfter

Zu Frage 1:

Anders als die uns vertrauten Grundrechenarten Addieren, Subtrahieren, Multiplizieren und Dividieren werden Undieren und Odern erst verständlich, wenn wir unseren Betrachtungen das binäre Zahlensystem zugrunde legen. Es existieren dort nur die Ziffern Eins und Null. Eine Aussage kann entweder wahr sein oder falsch. Eine wahre Aussage erhält den Wahrheitswert Eins (True), eine falsche Aussage den Wahrheitswert Null (False). Mit bestimmten Befehlen kann der Computer, je nachdem, ob eine Aussage zutrifft oder nicht, Anweisungen ausführen oder unterlassen. Dazu ein Beispiel:

```
1 IF 1 THEN PRINT
  "RICHTIG"

1 IF 0 THEN PRINT
  "RICHTIG"
```

Im ersten Fall erscheint eine Bildschirmausgabe, im zweiten Fall nicht.

Manchmal sollen mehrere Aussagen miteinander verknüpft werden. So soll der Computer eine Anweisung ausführen, wenn zwei Aussagen zutreffen. Hierzu gibt es den Operator AND. Nur wenn beide Aussagen wahr sind, ist auch deren Verknüpfung mit AND wahr.

```
PRINT 0 AND 0
PRINT 0 AND 1
PRINT 1 AND 0
PRINT 1 AND 1
```

Soll eine Anweisung ausgeführt werden, wenn eine Bedingung bereits wahr ist, steht uns der Operator OR zur Verfügung.

```
PRINT 0 OR 0
PRINT 0 OR 1
PRINT 1 OR 0
PRINT 0 OR 1
```

Wenn eine Aussage wahr ist, so ist auch deren Verknüpfung mit OR wahr.
Vielleicht haben Sie sich bereits gewundert über die ungewohnte IF-Anweisung:

```
IF A THEN PRINT
  "RICHTIG"
```

Der Computer erwartet nach IF nur einen Zahlenwert. Auch durch Vergleichsoperatoren wird nur ein Zahlenwert erzeugt.

```
PRINT 5=5
PRINT 5=4
```

Der Computer gibt allerdings nicht den Wert Eins für die wahre Aussage aus, sondern eine negative Eins. Der Grund hierfür ist, daß der Computer mit Integerzahlen von zwei Bytes Länge arbeitet. Zwei Bytes entsprechen 16 Bits. Alle 16 Bits werden auf den Wert Eins gesetzt. Das ergibt die Zahl $FFFF in hexadezimaler Darstellung. Werte von $0000 bis $7FFF werden positiv dargestellt, Werte von $8000 bis $FFFF werden negativ dargestellt, $FFFF als –1 und $8000 als –32768. Die IF-Anweisung sieht alle Werte, die ungleich Null sind, als richtig an. Für Verknüpfungen von Aussagen sollte aber ein einheitlicher Wert Verwendung finden, am besten –1, da zwei durch AND verknüpfte Werte, die ungleich Null sind, als Resultat Null liefern können.

Beispiel:

```
PRINT 2 AND 1
```

Die IF-Anweisung dürfte hiermit wohl eingehend erläutert sein, die AND- und OR-Operatoren und das binäre Zahlensystem allerdings zu wenig. Im C16/P4-SPECIAL Nr. 3 finden Sie mehr über Zahlensysteme. Der Artikel über Maschinensprache im selben Heft behandelt ausführlich OR- und AND-Befehle.

Zu Frage 2:

Soll eine längere Formel, die öfter im Programm benötigt wird, Verwendung finden, so bieten sich zwei Möglichkeiten an, wenn diese Formel nicht jedesmal neu hingeschrieben werden soll. Die erste Möglichkeit wäre, sie mit einem GOSUB anzuspringen, die zweite, sie mit DEF FN zu definieren. Dies geschieht in der Form:

```
1 DEF FN Y(X)
  =SQR(R*R-X*X)
```

Der DEF FN-Anweisung muß der Variablenname folgen mit in Klammer eingeschlossenem Übergabeparameter.
Nach dem Gleichheitszeichen folgt die Formel, die in unserem Beispiel für einen Kreis mit dem Radius R den zu X gehörenden Y-Wert berechnet. Wir ergänzen das Listing durch:

```
2 R=1
```

Der Kreis besitzt den Radius eins. Die Berechnung kann jetzt mit der FN-Anweisung vorgenommen werden, und das sogar im Direktmodus

```
?FN Y(0.5)
```

Der zu 0.5 gehörende Y-Wert ist 0.866, wie uns die Ausgabe zeigt.

Zu Frage 4:

Die Frage wurde vorgezogen, da erst nach dieser Erklärung auf Frage drei eingegangen werden kann. Steht hinter einer Variablen eine Klammer, in der sich vielleicht eine weitere Variable befindet, so ist dieses keineswegs eine Zuweisung. Es handelt sich hier um eine indizierte Variable. Betrachten wir uns einmal die normalen Variablen B1, B2, B3, B4. Nehmen wir ferner an, aus DATA-Zeilen sollten Werte eingelesen und den Variablen zugewiesen wer-

den. Es könnte sich natürlich auch um INPUT-Eingaben handeln. Viermal müßten wir die READ-Anweisung ausführen. Wenn es mehr Werte und Variablen wären, vielleicht 50, so hätten wir ganz schön viele Zeilen zu schreiben.
Eine Lösung sind die indizierten Variablen B(1), B(2), B(3) und B(4). Statt die Zahlen Eins bis Vier jedesmal dahinter zu schreiben, können wir diese auch durch eine andere Variable oder eine Formel ausdrücken. Damit ist es möglich, Wertzuweisungen in einer Laufschleife vorzunehmen:

```
1 FOR I=1 TO 4
2 READ B(I)
3 NEXT
```

Bei 50 Zuweisungen lassen sich viele Programmzeilen auf diese Weise sparen. Variablen können auch mehrfach indiziert sein.

INPUT A$(I,J)

Eine Anwendung hierfür wäre eine Datei mit mehreren Datenfeldern und Datensätzen. So würde hier vielleicht eine Eingabe in Datenfeld I im Datensatz J erfolgen oder umgekehrt, je nachdem, wie I und J im Programm verwendet werden.

Zu Frage 3:

Indizierte Variablen können nur verwendet werden, wenn der Index den Wert zehn nicht überschreitet. Für größere Index-Werte muß eine Dimensionierung stattfinden, damit der Rechner für diese Variablen Platz schaffen kann. Man spricht in diesem Zusammenhang auch von Datenfeld.

DIM A(19,29)

Dieses Datenfeld würde 20 mal 30 Felder = 600 Felder umfassen, zwanzig Felder in die eine Dimension und 30 in die andere. Räumlich vorstellen können wir uns dies als Liste mit 20 Spalten und 30 Zeilen, in die wir etwas hineinschreiben können. Mit der DIM-Anweisung bestimmen wir die Abmessungen (Dimensionen – der Liste.

KAUM ZU GLAUBEN

Vor einiger Zeit verscherbelte ein bekannter Discounter den C16 mit BASIC-Lehrkassette. Ich griff zu und war soweit zufrieden. Dann kam der Wunsch, über einen größeren Speicherbereich zu verfügen. Nach Anleitung habe ich dem C16 durch Austausch der beiden RAM-Bausteine den 64-KByte-Speicher-Bereich verpaßt.
Bekannterweise ist das Steckernetzteil des C16 nicht das beste. Irgendwann bin ich drangestoßen und der Bildschirm zeigte nur noch undefinierbare Zeichen – nichts ging mehr.
Verzweifelt schaltete ich den C16 (nun ja eigentlich C164) aus und an, immer wieder. Ich drückte die Reset-Taste, der Monitor wurde aus- und angeschaltet – doch nichts tat sich.
Also ist das Ding kaputt, weg damit in den Keller. Neuer Computer muß her.
So kam also der Plus/4 ins Haus, gleich mit Diskettenlaufwerk. Alles war wieder gut, denn die auf Kassette vorhandenen Programme laufen ja auch auf dem Plus/4.
Dann kam Ihre Zeitschrift C16/P4-SPECIAL mit dem an sich brauchbaren Beitrag „Screen-Keyboard".

Ich habe hiervon den ersten Teil verwendet. Mit der entsprechenden Änderung für die DIN-Tastatur und die im P/4 eingebauten Software.
Durch einen nicht mehr nachvollziehbaren Umstand drückte ich die Reset-Taste, nachdem das Screen-Keyboard-Programm im Rechnerspeicher stand.
Das Ergebnis war genauso umwerfend wie seinerzeit beim C16 – nichts ging mehr!
Aber welch ein Wunder – irgendwann ließ sich der Plus/4 doch zurücksetzen und war wieder zu gebrauchen.
Können Sie sich dies nicht nur vorstellen, sondern auch erklären?
Ich habe jedenfalls im Screen-Keyboard-Programm in Zeile 745 ein NEW eingefügt, und scheinbar ist dieses Problem keins mehr. Ich kann jetzt das besagte Programm laden, mit RUN starten und die Reset-Taste drücken – der P/4 folgt dieser Anweisung.
Aber es kommt noch besser: Heute hole ich mir den C16 (C164) aus dem Keller, schalte ihn ein – nichts. Wieder spiele ich den Turn durch – aus, an, aus, an und so weiter. Und mit einem Mal meldet sich der C16 (C164) mit seinen 60671 Bytes Free!
Das ist für mich schon mehr als mysteriös – ich ging bislang davon aus, daß das Ausschalten des Computers diesen in den „Urzustand" versetzt und nicht noch nach Monaten für Geisterbilder auf dem Monitor sorgt.
Sehen Sie eine Möglichkeit, mit vertretbarem Aufwand beide Computer so zu verbinden, daß entweder eine einfache Konfiguration eines Computernetzes entsteht oder daß zumindest beide Speicher so zusammenarbeiten können wie in der COMMODORE WELT 1/88, Seite 16, beschrieben (Umschalten von BANK 1 zu BANK 2)?
Ich habe den Gedanken, mit einem Monitor, einem Diskettenlaufwerk, einem Drucker, aber mit „einem Computer mit 2*64 KByte" zu arbeiten.
Reinhold Plennis
Bad Schwartau

Genauso mysteriös, wie Ihnen die Geschichte vorkommt, ist sie auch für uns. Ihre Annahme, daß nach dem Ein- und Ausschalten wieder der Anfangszustand vorliegen sollte, ist völlig richtig. Zwar gehen nach dem Ausschalten noch nicht alle Daten verloren, wie eine Überprüfung zeigt. Doch nach etwa zwanzig Sekunden sollten alle Speicherinhalte gelöscht sein. Das von Ihnen genannte Zeichensatzprogramm ist nicht resetfest, weshalb Sie den Rechner ohne weiteres durch einen Reset wieder in den Grundzustand hätten bringen sollen. Wir haben ebenfalls keinerlei Erklärung parat.
Es ist möglich, zwei Computer miteinander zu vernetzen. Leitungen sind genügend vorhanden. Was fehlt, ist nur ein Programm auf den beiden Rechnern, das den Datentransfer besorgt. Durch Austauschen der Speicherinhalte beider Rechner kann auch eine Art Bankumschaltung erfolgen. Da auch von anderer Seite Interesse über Informationsübermittlung zwischen mehreren Commodore-Computern geäußert wurde, wollen wir uns dieses Problems im folgenden Heft annehmen.

C16-P4-SPECIAL Nr. 3 *Seite 48*
ZEICHEN-CREATOR

In das Einleseprogramm schlichen sich aus ungeklärten Gründen Fehler ein. Am Ende der Zeile 8020 muß es heißen: GOTO 8040
Am Ende der Zeile 8030:
GOTO 8020

Computer-Fachbegriffe von a—z

dieser Ausgabe von PC POPULÄR

Absturz Durch falsche Bedienung oder Programm-Fehler kann es dazu kommen, daß ein Programm auf keine Eingabe mehr reagiert, also „abgestürzt" ist. Um die Funktionsfähigkeit wieder herzustellen, muß das Programm neu geladen werden. Ein Absturz ist meist auch mit Datenverlust verbunden und sollte bei guter Software ausgeschlossen werden können.

ASCII **A**merican **S**tandard **C**ode of **I**nformation **I**nterchange. 1963 entwickelter Standard-Code für Kleincomputer. Nur die ersten 128 Zeichen des Codes entsprechen der einheitlichen Norm. Die Belegung von 129 bis 255 kann je nach Hersteller variieren.

AT **A**dvanced **T**echnology, fortgeschrittene Technologie. Zusatzbezeichnung für einen leistungsstarken PC mit Intel 80286-, Intel 80386- oder Motorola 68xxx-Prozessor.

Batch-Datei In einer ausführbaren ASCII-Datei abgelegte Sammlung von Betriebssystem-Befehlen, die nach Aufruf der Datei automatisch nacheinander abgearbeitet werden. Batch-Dateien besitzen grundsätzlich die Erweiterung BAT.

BIOS **B**asic **I**nput/**O**utput-**S**ystem, Basis-Ein/Ausgabe-System. Steuert die Ein- und Ausgabe-Vorgänge des Rechners (Tastatur, Bildschirm, Schnittstellen). Wird nach dem Einschalten des Computers als erstes aus dem ROM geladen.

Bit **B**inary dig**it**, binäres Zeichen, kleinste ansprechbare Einheit eines Computersystems. Enthält entweder 0 oder 1 (Strom an/aus).

Bus Fachausdruck für parallel angeordnete Leiterbahnen, die einen Informationstransport zwischen verschiedenen Bauteilen, Baugruppen oder Peripheriegeräten gewährleisten.

Byte Kennwort für die Sammlung von acht binären Stellen zur Darstellung von 2 hoch 8=256 verschiedenen Bit-Kombinationen. Ein Byte enthält jeweils ein ASCII-Zeichen.

CASE **C**omputer **A**ided **S**oftware **E**ngineering. Programmentwicklung unter Zuhilfenahme des Computers als Entwicklungs- und Dokumentations-System, beispielsweise zur Erstellung von Struktogrammen mit Hilfe geeigneter Software.

Centronics Nach einem Drucker-Hersteller benannte parallele Schnittstelle, die inzwischen zum Standard geworden ist.

Chip Ein elektronisches Bauteil, das eine bestimmte Anzahl von Speicher- oder Schaltelementen enthält.

Co-Prozessor Schneller Prozessor für rechenintensive Aufgabenstellungen, der zusätzlich zum Haupt-Prozessor eingesetzt werden kann. Bei Personal-Computern als 8087 für XIs, 80287 für ATs und 80387 für die 386er-Rechner bezeichnet.

Code In der EDV Fachbegriff für Schlüssel, nach dem die Zeichen im Rechner abgelegt, erkannt und bearbeitet werden.

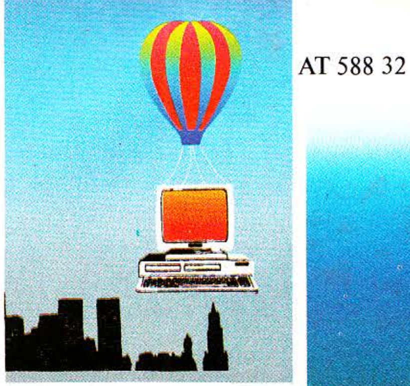
CPU **C**entral **P**rocessing **U**nit, Zentraleinheit, koordiniert die Abläufe im Rechner.

DIP-Schalter **D**ual **I**nline **P**ackage-Schalter. Eine Anzahl Kippschalter im Mini-Gehäuse (auch Mäuseklavier genannt) zum Umstellen der Parameter oder der werkseitigen Einstellungen bei Druckern, Computern, Erweiterungskarten, etc.

DOS Abkürzung für **D**isk **O**perating **S**ystem (Disketten-Betriebssystem). Siehe auch MS-DOS.

EDV Mittlerweile eingebürgerter Begriff für Elektronische Daten-Verarbeitung.

EGA **E**nhanced **G**raphics **A**dapter, verbesserter Grafik-Adapter (640 x 350 Pixel in 16/64 Farben). Verschiedene Arten dieser Karte (EGA-Wonder, EGA-HiRes) erreichen sogar noch eine höhere Auflösung.

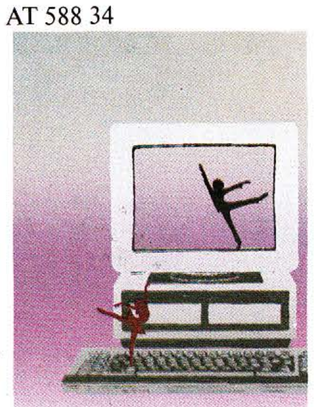


AT 588 31



AT 588 32

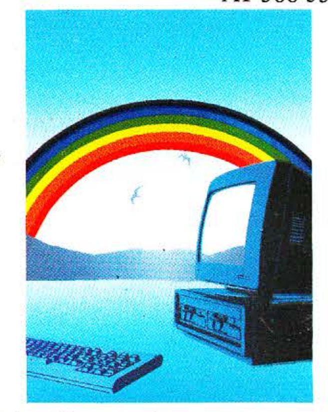
AT 588 33



AT 588 34

AT 588 29

Edition Computer-Kunst



AT 588 35

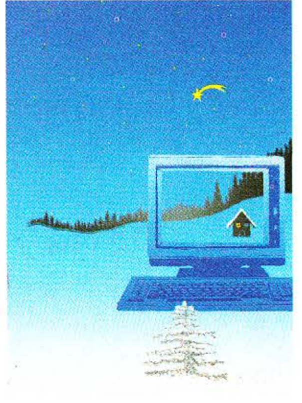


AT 588 30

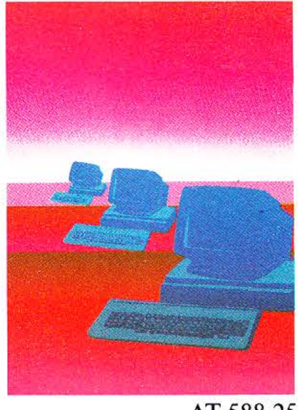
Computer sind nicht nur Gebrauchsgegenstände. Sie werden zunehmend auch von Künstlern als Motiv entdeckt. Einige besonders gut gelungene Arbeiten haben wir für Sie, unsere Leser, reservieren können. Alle Motive sind strikt auf eine Auflage von 99 Exemplaren limitiert und von der Künstlerin, Sybille Areco, handsigniert und numeriert. Die Exponate werden nach Bestelleingang im 24-Farbendruck von Hand gefertigt, die Vorlage nach dem 99. Druck vernichtet. Unser Angebot: Jedes Motiv nur DM 85,–, zwei Motive DM 150,–, drei DM 210,– und vier Motive nur DM 250,–. Jedes Bild ist 30 x 40 Zentimeter groß und kommt im Passpartout in stabiler Verpackung (im Preis enthalten).
Lieferzeit nach Bestelleingang: ca. drei Wochen.



AT 588 27



AT 588 28



AT 588 25



AT 588 26

Bestellcoupon

Hiermit bestelle ich in Kenntnis ihrer Verkaufsbedingungen folgende Exponate:

Nr.: ____________________

____

Ich zahle: (Zutreffendes bitte ankreuzen!)
per beigefügtem Scheck ☐ **Schein** ☐ **Gegen Bankabbuchung am Versandtag** ☐

Meine Bank (mit Ortsname) ____________

Meine Kontonummer ____________

Meine Bankleitzahl ____________ **(steht auf jedem Bankauszug)**

Nachname ____________ **Vorname** ____________

PLZ/Ort ____________ **Str./Nr.** ____________

Verkaufsbedingungen: Lieferung nur gegen Vorkasse oder Bankabbuchung.
Wichtig: Scheckeinreichung und Bankabbuchung erfolgen erst nach dem Versand. Keine Nachnahme möglich. Auf Wunsch Rechnung mit ausgewiesener Mehrwertsteuer.

Unterschrift ____________
Bitte ausschneiden und einsenden an

AKTUELL-VERLAG
Heßstraße 90
8000 München 40

File Engl. Bezeichnung für Datei.

GEM **G**raphics **E**nvironment **M**anager. Bezeichnet die grafische Benutzeroberfläche von Digital Research.

Hardcopy Die Ausgabe des Bildschirmspeichers auf einen Drucker. Der komplette aktuelle Bildschirm wird ausgedruckt. Schneller Aufruf über die Tasten Shift-PrtSc.

Interface Schnittstelle, Port. Anschlußpunkt für periphere Geräte am Rechner, beispielsweise Drucker, Modems oder Meß- und Überwachungsgeräte.

ISDN **I**ntegrated **S**ervices **D**igital **N**etwork, einheitliches öffentliches Netzwerk der deutschen Bundespost zur Übertragung digitaler Daten. Durch die Digitalisierung der Daten können gleichzeitig Telefongespräche, Computerdaten und Bilder übermittelt werden.

kB **k**ilo**B**yte, Abk. für 1000 (10 hoch 3). Bei dieser Abkürzung treten häufig Verwechslungen auf. Je nach Schreibweise des Anfangsbuchstabens ist ein anderer Wert gemeint.

Keyboard Engl. Bezeichnung für die Tastatur.

Laptop Batteriebetriebener PC, vollwertiger, handlicher ‚Schoßcomputer' und tragbarer Personal-Computer für unterwegs.

Mailbox Elektronischer Briefkasten, Sammelpunkt für private und öffentliche Nachrichten, die über das Telefonnetz zwischen verschiedenen Computersystemen ausgetauscht werden können.

MB **M**ega**B**yte, 1 MB = 1024 KB (2 hoch 20).

Motherboard Hauptplatine eines Rechners.

MS-DOS **M**icro**S**oft **D**isk **O**perating **S**ystem. Von Microsoft entwickeltes Betriebssystem für PC, das von Diskette oder Harddisk geladen wird. Allgemein als Standard anerkannt.

OS/2 **O**perating **S**ystem 2, neues leistungsstarkes Betriebssystem für Personal-Computer. In Zusammenarbeit zwischen Microsoft und IBM entwickelt.

parallel Art der Datenübertragung, bei der gleichzeitig 8 Bit (1 Byte) über verschiedene Leitungen nebeneinander übermittelt werden (siehe seriell). Als Standard hat sich die Centronics-Schnittstelle durchgesetzt.

Port Lat. „porta" für die „Pforte", siehe Interface.

Puffer Flüchtiger Zwischenspeicher zum Sammeln von Daten, die anschließend paketweise übertragen werden.

Pulldown-Menü Bezeichnet eine Fenstertechnik, bei der einzelne Menüs bei Bedarf aus einer einzelnen Zeile wie ein Rolladen heruntergezogen werden und neue Auswahlmöglichkeiten eröffnen.

Root Engl. für Wurzel, Ursprung. Bezeichnet das übergeordnete Hauptverzeichnis eines externen Speichermediums.

Schnittstelle siehe Interface.

Seriell Art der Datenübertragung, bei der jedes Byte in acht Bit zerlegt und nacheinander durch eine einzige Leitung übertragen wird (siehe parallel).
Zu den seriellen Schnittstellen gehören unter anderem die RS 232, RS 432 und die V-24.

Software Engl. für „weiche Ware", allgemeine Bezeichnung für Programme. Enthält eine, von Programm zu Programm unterschiedliche Anzahl von Befehlen, die zur Durchführung einer bestimmten Aufgabe vom Rechner der Reihe nach abgearbeitet werden.

Struktogramm Von Nassi und Ben Shneiderman entwickelte grafische Darstellung des Datenflusses in Programmen unter Verwendung von einheitlichen Symbolen.

Subdirectory Engl. für Unterverzeichnis. Dient auf einem Speichermedium zum Anlegen und Sammeln von Daten und Programmen verschiedener Anwendungsgebiete in voneinander unabhängigen Verzeichnissen (directories). Erhöht die Übersichtlichkeit und Zugriffsgeschwindigkeit.

Support Unterstützung, Hilfeleistung durch den Händler oder Hersteller nach dem Verkauf eines Gerätes oder Programmes.

Terminal „Unintelligentes" Datenendgerät zur Datenein-und -ausgabe, das über Leitungen mit einem Großrechner verbunden ist. Im Gegensatz zu einem Personal-Computer ist ein Terminal alleine nicht einsatzfähig.

Tools Engl. für „Werkzeuge", siehe Utility in Heft 8/88.

Wild Cards Ähneln in ihrer Handhabung den Jokern beim Kartenspiel. Sie können jedes beliebige Zeichen aufnehmen. Äußerst hilfreich vor allen Dingen auf der Betriebssystemebene und beim Suchen und Ersetzen. Als Jokerzeichen werden allgemein das Fragezeichen als Ersatz für ein einziges beliebiges Zeichen und der Stern "*" als Ersatz für mehrere zusammenhängende Zeichen verwendet.